Seeding with Costly Network Information

Dean Eckles[1], Hossein Esfandiari[2], Elchanan Mossel[3], M. Amin Rahimian[4]

[1]Sloan School of Management, MIT   [2]Google Research   [3]Department of Mathematics, MIT
[4]Department of Industrial Engineering, University of Pittsburgh
eckles@mit.edu, esfandiari@google.com, elmos@mit.edu, rahimian@pitt.edu

We study the task of selecting k nodes, in a social network of size n, to seed a diffusion with maximum expected spread size, under the independent cascade model with cascade probability p. Most of the previous work on this problem (known as influence maximization) focuses on efficient algorithms to approximate the optimal seed set with provable guarantees given knowledge of the entire network; however, obtaining full knowledge of the network is often very costly in practice. Here we develop algorithms and guarantees for approximating the optimal seed set while bounding how much network information is collected. First, we study the achievable guarantees using a sublinear influence sample size. We provide an almost tight approximation algorithm with an additive ϵn loss and show that the squared dependence of sample size on k is asymptotically optimal when ϵ is small. We then propose a probing algorithm that queries edges from the graph and use them to find a seed set with the same almost tight approximation guarantee. We also provide a matching (up to logarithmic factors) lower-bound on the required number of edges. This algorithm is implementable in field surveys or in crawling online networks. Our probing takes p as an input which may not be known in advance, and we show how to down-sample the probed edges to match the best estimate of p if they are collected with a higher probability. Finally, we test our algorithms on an empirical network to quantify the tradeoff between the cost of obtaining more refined network information and the benefit of the added information for guiding improved seeding strategies.

Key words: Influence maximization, submodular maximization, query oracle, viral marketing

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 an 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) (Banerjee et al. 2019, Domingos and Richardson 2001, Godes and Mayzlin 2009, Hinz et al. 2011, Kempe et al. 2003, Libai et al. 2013). The influence maximization problem is to choose a set of k seeds with maximum expected spread size, given a known network and model of diffusion (Domingos and Richardson 2001). Following the seminal work of Kempe et al. (2003) — who showed NP-hardness and efficient approximation through submodular influence

maximization — a huge literature is devoted to developing fast algorithms that can be applied to massive scale social networks (e.g., Chen et al. 2009, Wang et al. 2012).

In this work, we address the problem of influence maximization when the social network is unknown and so network information needs to be acquired through costly effort. This has applications in development economics — e.g., adoption of microfinance (Banerjee et al. 2013) and insurance (Cai et al. 2015); public health — e.g., adoption of water purification methods and multivitamins (Kim et al. 2015), spreading information about immunization camps (Banerjee et al. 2019), preventing misinformation about drug side effects (Chami et al. 2017), increasing HIV awareness among homeless youth (Yadav et al. 2017), and adoption of contraception (Behrman et al. 2002); and education — e.g., reducing bullying and conflict among adolescents (Paluck et al. 2016). In these settings, data about network connections is often acquired through costly surveys. In practice, collecting the entire network connection (edge) data can be difficult, costly, or even impossible. 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 (Chami et al. 2017, Chin et al. 2021, Kim et al. 2015). Moreover, such methods have the advantage of scalability, since they can be implemented without mapping the entire network. This is also important in online social networks with billions of edges, where working with the entire contact lists might be impractical or limited by rate limits for third parties crawling these networks. Although the importance of influence maximization with partial network information has been noted and there are a few papers considering this problem (Mihara et al. 2015, 2017, Stein et al. 2017, Wilder et al. 2018), none of these previous works come with provable performance guarantees for general graphs.

To limit access of seeding algorithms to network information, we use an edge query model and provide tight guarantees of what is achievable with a bounded number of queries. We organize our edge queries by sequentially probing the graph nodes: we probe each node by revealing its incident edges with independent cascade probability p, proceed to probe its revealed neighbors, and repeat. Our approximation algorithm uses the revealed network information to seed k nodes with guarantees that match hardness lower bounds (up to logarithms).

We begin our analysis by a thought experiment (Section 2.1): assuming that network information is made available through "influence samples", i.e., by seeding random nodes and observing their spread outcomes, how many influence samples do we need to collect? We show that to seed k nodes in a network of size n with tight approximation guarantees, it is necessary (up to logarithms) and sufficient to collect $O(k^2 \log n)$ influence samples. In Section 3, we provide our main results by showing that the same approximation guarantees can be achieved using $O(pn^2 \log^4 n)$ edge queries (with a matching lower bound). Our probing mechanism for edge queries makes use of the independent

cascade probability p to sample edges; therefore, in subsection 3.5, we study what happens when the probe and seed cascade probabilities (denoted by p' and p, respectively) are different. We point out the hardness of giving general guarantees when $p' \neq p$ and propose a post-processing solution to correct for this discrepancy as long as $p' > p$, i.e., the edge data are collected with sufficiently high probability. In Section 4, we use our bounded-query framework to resolve a trade-off between the cost of acquiring network information and its benefit in increasing expected spread size. We provide discussion and concluding remarks in Section 5. Detailed comparisons with related works are provided in Appendix A. Detailed proofs are presented in Appendix B. In Appendix C, we discuss the extension of our results to other influence models including independent cascade on directed graphs (Appendix C.1) and the linear threshold model, for which we provide approximation guarantees using only $O(nk^2 \log n)$ edge queries (Appendix C.2).

1.1. Main contributions

We consider the independent cascade (IC) model of social contagion that is fairly well-studied since its use by Kempe et al. (2003). In this model, network edges are "active" with probability p independently of each other and all nodes with active connections to other active nodes become active. Motivated by applications to product and technology adoption, we refer to active nodes as adopters. Starting from a set of initial adopters, the adoption propagates through the network and the process terminates after a finite number of steps. Following the independent cascade model, every adopter has a single chance to activate each of its neighbors independently with probability p. The k-influence maximization problem, or k-IM in short, refers to the choice of k initial adopters to maximize expected adoptions under this diffusion model. Let OPT 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 μOPT. In this work, we assume a query oracle access to the network graph and study the k-IM problem with a limited number of queries.

We begin with a hypothetical scenario assuming that we can pay a cost to seed a random node and learn the outcome of the spreading process (e.g., imagine distributing traceable coupons to random individuals and asking them to pass the coupons to their friends; or a social network marketing firm that measures its audience by seeding ads and promotional goods randomly). We only learn the identity of the final adopters and do not use any information about the network edges through which the influence spreads. We collect several independent cascade outcomes by repeating this process and refer to them as "influence samples". We use these influence samples to seed k nodes with optimality guarantees. We first show that an additive loss (e.g., ϵn) is necessary, given $o(n)$ influence samples (Theorem 1, Subsection 2.1):

HARDNESS OF APPROXIMATION WITH $o(n)$ INFLUENCE SAMPLES. *Let $\mu > 0$ be any constant. There is no μ-approximation algorithm for influence maximization using $o(n)$ influence samples.*

Interestingly, we show that $O(k^2 \log n)$ influence samples are enough to provide a k-IM solution with almost tight approximation guarantees. For example, if finding a single seed on a star ($k = 1$), with high probability all random samples are leaves of the star. However, based on the $O(\log n)$ spread outcomes our algorithm finds and seeds the center of the star. We also show that the quadratic order dependence on k is the best possible. The following is a formal summary of our results from Theorems 2 and 3 in Subsection 2.1:

APPROXIMATION GUARANTEES WITH BOUNDED NUMBER OF INFLUENCE SAMPLES. *For   any arbitrary $0 < \epsilon \leq 1$, there exists a polynomial-time algorithm for k-influence maximization that covers $(1 - 1/e)\mathsf{OPT} - \epsilon n$ nodes in expectation using no more than $O_\epsilon(k^2 \log n)$ influence samples. Moreover, there can be no approximation algorithms that provide $\mu\mathsf{OPT} - \epsilon n$ guarantees for k-IM using $o(k^2)$ influence samples for a fixed $0 < \mu < 1$ and $0 < \epsilon < \mu/k$.*

Notice that our bound on the number of influence samples depends logarithmically on n, therefore, when k is poly-logarithmic we only use poly-logarithmic number of influence samples which is exponentially lower than the best known bound of $O(kn \log n)$ for sample complexity of influence maximization on general graphs (Sadeh et al. 2020, Section 2). We point out that our order n improvement is only possible because we allow for an additive loss in our approximation guarantee. Detailed comparisons with this and other related works are presented in Appendix A.

Our main contribution is to show that similar approximation guarantees are possible as we bound the total number of edge queries, i.e., queries of the form (v, i) that return the i-th neighbor of node v with arbitrarily ordered neighborhoods. We propose a probing procedure to sequentially reveal random neighborhoods of the nodes, resulting in a snowball-like sampling of the network edges. Notice that *a single* simulation of the independent cascade model over the entire network (without using our subsampling and stopping constraints) requires $\Omega(pn^2)$ edge 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 the optimum (Theorem 4, Section 3):

HARDNESS OF APPROXIMATION WITH $o(n^2)$ EDGE QUERIES. *Let μ be any constant. There is no μ-approximation algorithm for influence maximization using $o(n^2)$ edge queries.*

We avoid the above impossibility by allowing for an ϵn additive loss in our approximation guarantee. Subsequently, one natural question that arises 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 positively by showing that our probing scheme approximately preserves the greedy solution to the k-IM problem, achieving a $(1 - 1/e)\mathsf{OPT} - \epsilon n$ guarantee using no more than $O_\epsilon(pn^2 \log^4 n + \sqrt{kp}n^{1.5} \log^{5.5} n + kn \log^{3.5} n)$ edge queries. We also provide a matching lower bound (up to logarithms) to show that the linear order dependence on p is tight. The following is a formal restatement of our results in Theorems 6 and 7 of Subsection 3.4.

APPROXIMATION GUARANTEES WITH BOUNDED NUMBER OF EDGE QUERIES. *For any arbitrary $\epsilon > 0$, there exists a polynomial-time algorithm for influence maximization that covers $(1 - 1/e)\mathsf{OPT} - \epsilon n$ nodes in expectation, using $O_\epsilon(pn^2 \log^4 n + \sqrt{kp}n^{1.5} \log^{5.5} n + kn \log^{3.5} n)$ queries, where OPT is the expected number of nodes covered by the optimum solution to k-IM. Moreover, there can be no approximation algorithms that provide $\mu\mathsf{OPT} - \epsilon n$ guarantees for k-IM using $o(pn^2)$ edge queries for a fixed $0 < \mu \le 1$ and $\epsilon < \mu^2/18$.*

To achieve this result, we apply some subsampling techniques with stopping constraints that enable us to *approximately simulate* $O_\epsilon(k \log n)$ independent cascades, starting from a random sample of $O_\epsilon(k \log n)$ initial nodes and using only $O_\epsilon(pn^2 \log^4 n + \sqrt{kp}n^{1.5} \log^{5.5} n + kn \log^{3.5} n)$ edge queries. We specify the dependence of the O_ϵ terms on ϵ when presenting our main results in Section 3. Of note, our subsampling technique makes critical use of the independent cascade probability p when deciding how many edges to query in the neighborhood of each node. In practice, the true value of p is often subject to significant uncertainty. We address the dependency of our edge queries on p with a hardness result in Section 3.5 and discuss how potential discrepancies may be corrected if p is unknown at data collection but measurable afterwards.

The most closely related result that provides approximation guarantees for k-IM with limited queries to an unknown graph is due to Wilder et al. (2018), who propose an algorithm for input graphs that are drawn from a particular family of stochastic block models. Their algorithm, which is tailored to that specific random graph model, 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 $TR \in O(\log^6 n)$ nodes. The k nodes in the seed set are selected from the initial T samples, such that the k largest blocks are seeded uniformly at random. Unlike Wilder et al. (2018), we do not make any assumptions about inputs, so our results are applicable to general graphs. In the following subsection, we put our contributions in perspective by discussing related bodies of literature. Detailed discussions of methodologically relevant work are provided in Appendix A. Our main results in Section 3 are presented for undirected graphs. In Appendix C.1, we provide the extension of the edge query model to directed graphs and show that the same approximation guarantees and query bounds hold true when nodes are queried for their influencers (their incoming edges).

1.2. Related work

Motivated by the difficulties of acquiring complete network data, we are interested in methods for targeting in networks without making explicit use of the full graph. Such methods have roots in multiple applied problems — vaccination (Cohen et al. 2003) and disease surveillance (Christakis and Fowler 2010) — in addition to seeding. One approach that has received substantial attention is a "one-hop" strategy [sometimes called "nomination" (Kim et al. 2015) or "acquaintance targeting" (Chami et al. 2017, Cohen et al. 2003)] that selects as seeds the neighbors of random nodes. This approach exploits a version of the friendship paradox that states: "the friend of a random individual is expected to have more friends than a random individual," (Feld 1991, Lattanzi and Singer 2015). For example, Kim et al. (2015) report on the results of field experiments that target individuals for delivery of public health interventions (spreading adoption of multivitamins and a water purification method). For one product, they argue 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 (Chin et al. 2021, cf. Kumar and Sudhir 2019). While there are results about how these short random walks affect the degree distribution of selected nodes (Kumar et al. 2018), 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 ask 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. In related work, Akbarpour et al. (2020) study the value of network information for seeding interventions. We provide a detailed comparison with this work in Section 4, after clarifying our modeling assumptions and results.

In another related work, Manshadi et al. (2020) 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).

Particularly relevant to our present study is recent work on influence maximization for unknown graphs (Mihara et al. 2015, 2017, Stein et al. 2017, Wilder et al. 2017a, 2018). Mihara et al. (2015) use a biased snowball sampling strategy to greedily probe and seed nodes with the highest degree; they later propose to improve their heuristic by including random jumps that avoid excessive local search in their snowball sampling strategy (Mihara et al. 2017). Stein et al. (2017) explore applications of common heuristics and 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. (2018). However, as discussed above (section 1.1), this algorithm and analysis is tailored to graphs generated from a particular family of stochastic block models (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.

We rely on sketching techniques to summarize influence functions using a bounded number of queries; in Subsection 3.3, we adopt high-level ideas from Bateni et al. (2018, 2017) for construction of our sketch (see Lemma 4). Cohen et al. (2014) also develop a sketch-based algorithm for influence maximization to bound the running time with approximation guarantees. In other related work, Borgs et al. (2014) give a quasi-linear time algorithm for influence maximization based on reversed influence samples that use $O(k(n+m)\log n)$ edge queries, where m is the size of the input edge set. Although these algorithms achieve fast (nearly best possible) influence maximization, they may query all edges multiple times on some inputs because they are not directly concerned with limiting query access to unknown graphs. In Appendix A, we give a detailed comparison with these and other relevant works that are based on (reverse) influence sampling.

Our influence maximization guarantees also relate to the recent developments in stochastic submodular maximization (Karimi et al. 2017), as well as optimization from samples (Balkanski et al. 2017a, 2016). The key difference is that in our algorithms we make explicit use of the combinatorial structure of the collected data. This is in contrast to the optimization from samples framework, 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 limitations of the optimization from samples (cf. Balkanski et al. 2017b). We provide more details about our relationship with this literature in Appendix A.

Some prior work has addressed lack of perfect knowledge about the spreading process by learning the influence model and potentially heterogeneous probabilities of spreading along each edge (Gomez-Rodriguez et al. 2012, Goyal et al. 2010). However, rather than attempting to learn the

model parameters from existing data, we are interested in data collection as an active, costly process that is performed to inform seeding interventions. To this end, we offer a methodology that coordinates data collection and influence maximization by limiting queries to the social network graph. In other online learning and bandit-based approaches, the learner can select different seed sets at each stage and receives feedback from seeding in previous stages (Wen et al. 2017, Wu et al. 2019). This is also related to adaptive seeding, where the initial choice of seeds influences what becomes available for seeding in a followup stage (Feng et al. 2020, Horel and Singer 2015, Seeman and Singer 2013). The ability to seed nodes adaptively makes such setups incomparable to ours — and inapplicable when practical considerations demand commencing seeding simultaneously.

2. Problem setup and preliminary results

Consider a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with the set of nodes $\mathcal{V}$, the set of edges $\mathcal{E}$ and a seed set $\mathcal{S} \subseteq \mathcal{V}$. Starting from the seeded nodes in $\mathcal{S}$, adoption spreads along the edges of $\mathcal{E}$ with independent cascade probability p according to the IC model in Section 1.1. 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. The influence function, Γ, maps each seed set, $\mathcal{S}$, to its value, $\Gamma(\mathcal{S}) = \sum_{v \in \mathcal{V}} \phi(v, \mathcal{S})$, 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 $\mathcal{S} \subset \mathcal{V}$ with $card(\mathcal{S}) = k$ to maximize $\Gamma(\mathcal{S})$. We use $\Lambda = \arg\max_{\mathcal{S}, card(\mathcal{S}) = k} \Gamma(\mathcal{S})$ to denote any such solution and use $\mathsf{OPT} = \Gamma(\Lambda)$ to denote the optimal value.

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

An important result in influence maximization is that Γ is a non-negative, monotone, submodular set function (Kempe et al. 2003, 2005, 2015, Mossel and Roch 2010). Subsequently, it can be approximately maximized by sequentially selecting k seeds with the largest marginal gains, i.e., the greedy algorithm, which makes $O(nk)$ oracle calls to Γ and achieves a $1 - (1 - 1/k)^k \geq (1 - 1/e)$ approximation guarantee (Nemhauser et al. 1978). The greedy algorithm sets a gold standard for influence maximization that is NP-hard to improve upon — indeed, k-IM generalizes the maximum coverage problem and suffers its hardness of approximation beyond a $1 - (1 - 1/k)^k$ factor. Here, we achieve *roughly* the same guarantee without oracle access to Γ and using only a limited number of queries to 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 $\alpha\mathsf{OPT} - \epsilon n$. We can thus achieve similar worst-case guarantees using only partial information about the network.

Accessing the input graph by performing edge queries is a common technique in sublinear time algorithms that inspect only a small portion of their input before providing an output (Alon et al. 2000, 2009, Chazelle et al. 2005, Esfandiari and Mitzenmacher 2018, Indyk 1999). Formally, we assume that the input graphs are represented by an adjacency list defined as a collection of lists, $\{(\mathcal{N}_\nu, \mathrm{card}(\mathcal{N}_\nu)), \nu \in \mathcal{V}\}$, where each list, $\mathcal{N}_\nu$, consists of all the neighbors of node ν in some arbitrary (but fixed) order, and is accompanied by its length. Our query oracle model is defined such that given a vertex ν and an index $1 \le i \le \mathrm{card}(\mathcal{N}_\nu)$, the algorithm can query who is the i-th neighbor of ν (Gonen et al. 2011):

DEFINITION 3 (EDGE QUERY). Given a vertex $\nu \in \mathcal{V}$ and an index $i \in \{1, \ldots, \mathrm{card}(\mathcal{N}_\nu)\}$, an edge query with parameters (ν, i) reveals the i-th neighbor of ν.

We use edge queries as part of a probing mechanism, whereby a node is asked to reveal her neighbors each with probability p. Formally, probing node ν is defined as follows:

DEFINITION 4 (PROBING). Given a vertex $\nu \in \mathcal{V}$, a probing with parameters (ν, p) performs a sequence of (ν, i) edge queries for every index i that remains after eliminating the elements of the index set $\{1, \ldots, \mathrm{card}(\mathcal{N}_\nu)\}$, independently at random, with probability $1 - p$.

Of note, the total number of edge queries that are performed as a result of a (ν, p) probing is a binomial random variable with size parameter $\mathrm{card}(\mathcal{N}_\nu)$ and success probability p. In addition to probing nodes and running edge queries, our algorithm also needs a subsample of initial nodes that are chosen at random without replacement from the node set. Given this initial sample, we repeatedly probe the extended neighborhoods of the initial sample using edge queries. In our analysis, we bound the total number of edge queries that our algorithm makes in order to achieve the desired approximation guarantees. We also provide a query complexity lower-bound to show that the probing algorithm is order optimal for achieving the desired approximation guarantees, with as few queries as possible.

Although accessing the input graph locally by revealing the ordered neighborhoods of its nodes is a common query oracle model, our probing setup is also motivated by practical methods of network sampling such as snowball sampling, link-tracing, and respondent-driven sampling (RDS) that are popular in public health surveillance, social policy research, sociology, and survey design applications (Heckathorn and Cameron 2017). The principle utility of such methods is in constructing samples of hidden (hard to reach) populations (e.g., when estimating prevalence of HIV among drug injectors). In these situations, research begins with a convenience sample of initial subjects which is then
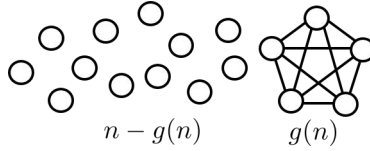
Figure 1 Using $f(n) \in o(n)$ influence samples on a graph comprised of a clique of size $g(n) = \sqrt{n/f(n)}$ and $n - g(n)$ isolated nodes, one cannot achieve an approximation factor that is better than $o(1)$.

expanded by tracing their network links in waves, until the target sample size is attained (Heckathorn and Cameron 2017, Salganik and Heckathorn 2004, cf. Goel and Salganik 2010). Following our probing procedure, researchers can decide which links to trace in the neighborhood of a probed node, randomly by simulating independent biased coin flips with head probability p. More broadly, our PROBE algorithm (Algorithm 2, Section 3) can be integrated with social network data collection software — e.g., the Trellis mobile platform (Lungeanu et al. 2021) used in Kim et al. (2015) and other studies — to generate survey sampling plans for researchers in the field.

2.1. Approximation guarantee with a bounded number of influence samples

To demonstrate the challenges of seeding with partial network information, we present a thought experiment whereby one can pay a cost to learn the outcome of a spreading process when a single node is seeded. Imagine giving out coupons (or lottery tickets) to random individuals and observing their usage spread as a way of collecting network information. Formally, we define an "influence sample" as the outcome of seeding a random node:

DEFINITION 5 (INFLUENCE SAMPLE). Each influence sample consists of all nodes that become active, after a single node is chosen uniformly at random (with replacement) and seeded.

As a theoretical exercise, we ask how many influence samples we need to collect to be able to provide a k-IM approximate solution. We first show that one cannot hope to provide a constant factor approximation guarantee, μOPT, for any $\mu > 0$, using $o(n)$ influence samples. Our hard example consists of a graph with a small clique and many isolated nodes (Figure 1). In such a structure, using $o(n)$ influence samples one is unlikely to observe the small clique and cannot achieve better than an $o(1)$ approximation factor. This example is similar to one in Wilder et al. (2018, Theorem 1), but we improve their $O(n^{1-\epsilon})$ lower bound to $o(n)$. The proof details are in Appendix B.1.

THEOREM 1. *Let $0 < \mu < 1$ be any constant. There is no μ-approximation algorithm for influence maximization using $o(n)$ influence samples.*

Knowing that a multiplicative approximation guarantee is impossible with $o(n)$ influence samples, we next ask how many influence samples we need for providing a $(1 - 1/e)\text{OPT} - \epsilon n$ guarantee with fixed $\epsilon > 0$. Algorithm 1 provides such a guarantee using $k\rho \in O_\epsilon(k^2 \log(nk))$ influence samples,

where $\rho = \rho_\epsilon^{n,k} = \lceil (81k/\epsilon^3) \log(6nk/\epsilon) \rceil$ is the number of influence samples we collect to choose one seed. The total number of influence samples that we use in Algorithm 1 is

$$T_\epsilon^{n,k} = k\rho_\epsilon^{n,k} = k \left\lceil \frac{81k \log(6nk/\epsilon)}{\epsilon^3} \right\rceil \in O\left(\frac{k^2 \log n}{\epsilon^3} + \frac{k^2 \log(1/\epsilon)}{\epsilon^3} \right). \tag{1}$$

Algorithm 1: INF-SAMPLE(ρ, k)

Input: Influence sampling access to graph $\mathcal{G}$, sample size ρ, and seed set size k

Output: $\Lambda^\star$, approximate seed set of size k with value at least $(1 - 1/e)\mathsf{OPT} - \epsilon n$

1 Initialize $\Lambda^\star \leftarrow \varnothing$.

2 for i **from** 1 **to** k **do**

3   Collect ρ influence samples and call them A_1^i, ..., A_ρ^i.

    `// Discard the influence samples that intersect with already chosen seeds (`$\Lambda^\star$`):`

4   **for** j **from** 1 **to** ρ **do**

5   **if** $A_j^i \cap \Lambda^\star \neq \varnothing$ **then**

6 $A_j^i \leftarrow \varnothing$.

7   **end**

8   **end**

    `// Choose the `i`-th seed based on the remaining influence samples:`

9   **for** $u \in \mathcal{V} \setminus \Lambda^\star$ **do**

10   **for** j **from** 1 **to** ρ **do**

11 $X_{u,j}^i \leftarrow \mathbb{1}\{u \in A_j^i\}$.

12   **end**

13 $X_u^i \leftarrow \sum_{j=1}^\rho X_{u,j}^i$.

14   **end**

15 $v^\star \leftarrow \arg\max_{u \in \mathcal{V} \setminus \Lambda^\star} X_u^i$.

16 $\Lambda^\star \leftarrow \Lambda^\star \cup \{v^\star\}$.

17 end

18 return $\Lambda^\star$.

The following theorem formalizes our guarantees for Algorithm 1. Its proof is in Appendix B.2. The main idea is that nodes that appear in many influence samples are good candidates for seeding since they are reached by many random nodes. For example, in Figure 2 the black node is the only node that appears in all influence samples and is the best candidate for seeding. To prevent overlap with the previously chosen seeds, at each step we discard those influence samples that contain any of the already chosen seeds (belonging to $\Lambda^\star$). The crux of the argument is in realizing that $(n/\rho)X_u^i$ is an unbiased estimator of the expected marginal gain from adding u to the seed set. By controlling
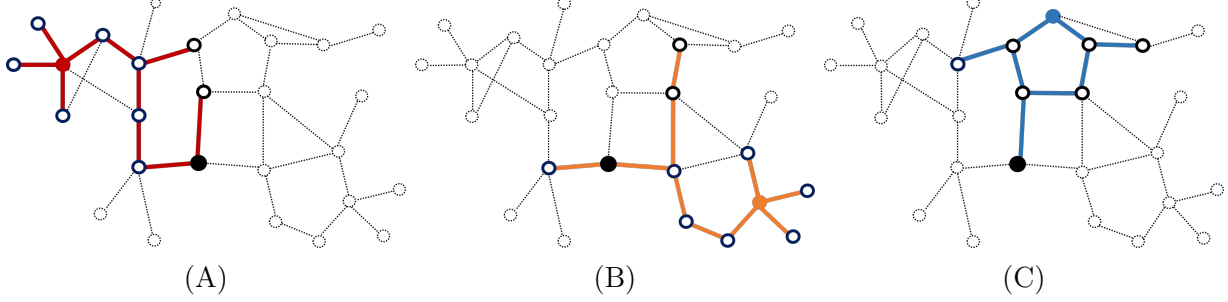
Figure 2   Three influence samples are depicted in (A) red, (B) orange, and (C) blue. In each influence sample, the random initial node is marked in the same color as the cascade. The node that appears most across different samples is marked in black. The dotted segments are not observed in the samples.
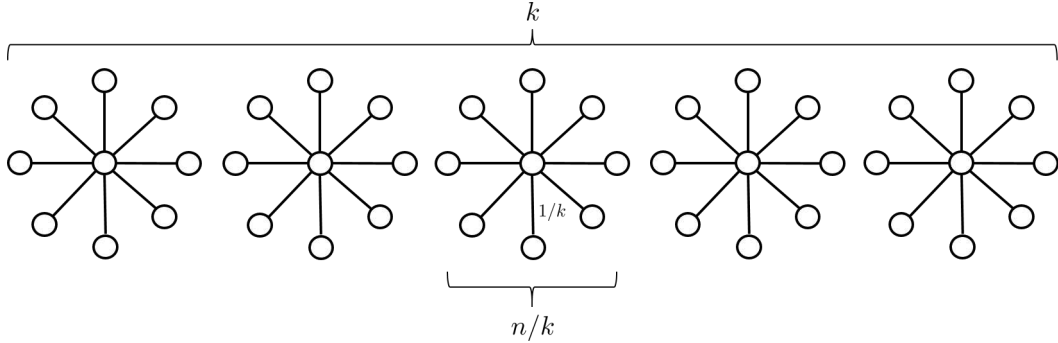

Figure 3   Using $o(k^2)$ influence samples on a graph comprised of k stars of size n/k each, with cascade probability $p = 1/k$, one cannot achieve a $\mu\mathsf{OPT} - \epsilon n$ approximation guarantee for $\epsilon < \mu/k$.

the deviation of X_u^i from $\mathbb{E}[X_u^i]$, we can approximate every step of the greedy algorithm by choosing u from $\mathcal{V} \setminus \Lambda^\star$ to maximize X_u^i.

THEOREM 2. *For any arbitrary $0 < \epsilon \leq 1$, there exists a polynomial-time algorithm for influence maximization that covers $(1 - 1/e)\mathsf{OPT} - \epsilon n$ nodes in expectation in $O(k^2 \log(n/\epsilon)/\epsilon^3)$ time, using no more than $k \lceil 81k \log(6nk/\epsilon)/\epsilon^3 \rceil \in O(k^2 \log(n/\epsilon)/\epsilon^3)$ influence samples.*

We end this section by a lower bound on the required number of influence samples for achieving the $(1 - 1/e)\mathsf{OPT} - \epsilon n$ approximation guarantee. In particular, we show that the k^2 asymptotic rate for $T_\epsilon^{n,k}$ in (1) is optimal for $\epsilon < \mu/k$. Our hard example consists of a collection of k stars of size n/k each, with independent cascade probability $p = 1/k$; see Figure 3. The optimum achieves $k(n/k^2) = n/k$ expected spread size by seeding the centers of each of the k stars. In Appendix B.3, we show that any algorithm that uses $o(k^2)$ influence samples can at most discover a small expected fraction of the center nodes, and therefore, fails to guarantee a $\mu\mathsf{OPT} - \epsilon n$ expected spread size when $\epsilon < \mu/k$.

THEOREM 3. *Fix $0 < \mu < 1$ and $0 < \epsilon < \mu/k$. There can be no approximation algorithms that provide $\mu\mathsf{OPT} - \epsilon n$ guarantees for k-IM using $o(k^2)$ influence samples.*
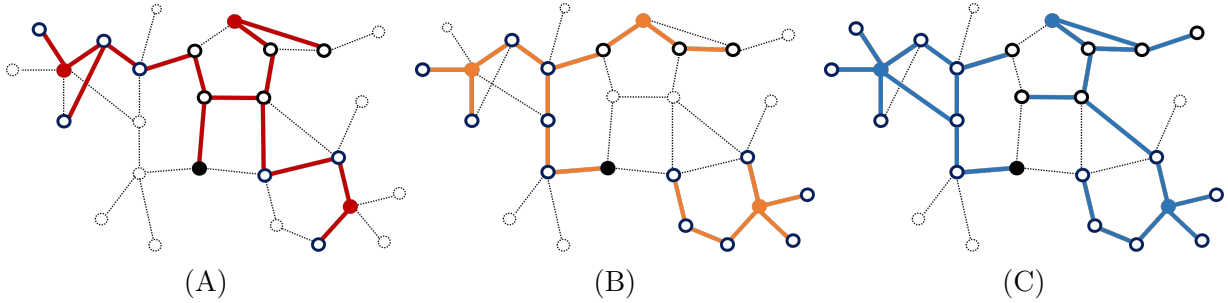
Figure 4 Three example cascades obtained through edge queries, depicted in (A) red, (B) orange, and (C) blue. All cascades start from the same random initial nodes which are marked in the same color as the cascades. The node that is marked in black scores as high as or higher than other nodes across the three cascades. The dotted sections consist of unsampled edges and nodes.

While these results characterize the challenges of seeding with limited network information, influence samples are often not a practical query model in a number of settings of interest and do not readily extend to directed graphs (Appendix Appendix C.1). Thus, we turn to another, more widely-applicable query method.

3. Approximation guarantees with bounded edge queries

In this section, we present an algorithm to perform edge queries by probing the extended neighborhood of a random subsample of the network nodes (Algorithm 2: PROBE), as well as an algorithm to output an approximate seed set, given the outcome of the edge queries (Algorithm 3: SEED). The main idea of the PROBE algorithm is to simulate multiple independent cascades starting from a set of random initial nodes; by querying each node about its neighbors and repeating the same for the revealed neighbors, neighbors of neighbors, etc. The SEED algorithm takes in the output of the PROBE algorithm and chooses seeds that are connected to the most initial nodes along the queried edges. In Figure 4, we depict an example of three cascades that are obtained through edge queries. Consider the node that is marked in black. Its queried connected component has three initial nodes in 4(A), two initial nodes in 4(B), and one initial node in 4(C). The value of each connected component is the number of initial nodes in that component and adding them gives the total value of the marked node: $1 + 2 + 3 = 6$. To prevent overlap with already chosen seeds, we set the value of a connected component to zero once one of its nodes is seeded. Using these valuations, the SEED algorithm approximates the greedy algorithm by sequentially adding the most valuable candidates to the seed set and updating the value of the connected components.

Our analysis consists of a $(1 - 1/e)\mathsf{OPT} - \epsilon n$ lower bound on the expected spread size of the chosen seed set, as well as upper bounds on the total number of edges that are queried by the PROBE algorithm and the subsequent run-time of the SEED algorithm. Before digging any further into the
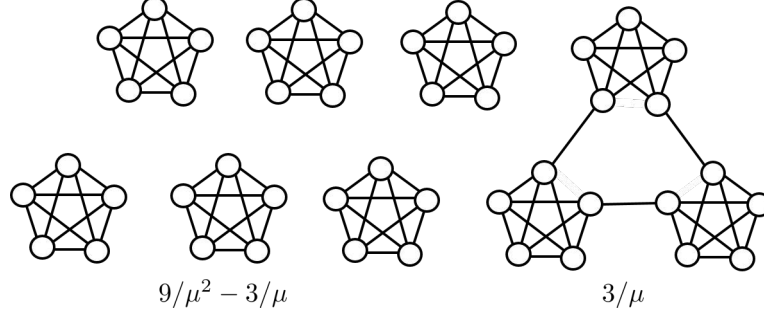
$$9/\mu^2 - 3/\mu \qquad\qquad\qquad 3/\mu$$

Figure 5   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.

analytical details, let us provide a companion hardness result that parallels Theorem 1 for influence samples, showing that a multiplicative approximation guarantee is, in general, not achievable using a nontrivial number of edge queries (thus the additive loss in our lower bound). We provide a hard example containing $(9/\mu^2)\binom{n\mu^2/9}{2}$ edges for which one cannot provide a μ-approximation while querying $o(n^2)$ edges. To this end, 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 B.4. 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 5. 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 B.4, we show that the expected spread size from seeding the output of any such algorithm is less than $n\mu^2/3$, i.e., less than a factor μ of the optimum.

THEOREM 4. *Let $0 < \mu \leq 1$ be any fixed constant. There can be no μ-approximation algorithm for influence maximization using $o(n^2)$ edge queries.*

The hallmark of our analysis is in identifying an auxiliary submodular function, $\Gamma_\delta : 2^{\mathcal{V}} \to \mathbb{R}$, to approximate our submodular function of interest $\Gamma : 2^{\mathcal{V}} \to \mathbb{R}$. 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, we use Λ_δ and Λ to denote the maximizers of Γ_δ and Γ with constrained size k. The following Lemma (proved in Appendix B.5) 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 1. *Consider set functions Γ_δ and Γ that map subsets of $\mathcal{V}$ to $\mathbb{R}$ with their respective maximum values, OPT_δ and OPT, on subsets of size k. Assume that for all seed sets of size k, $\mathcal{S}$, we have $|\Gamma_\delta(\mathcal{S}) - \Gamma(\mathcal{S})| \leq \epsilon n$. Let Λ'_δ be any approximate maximizer of size k for Γ_δ, satisfying $\Gamma_\delta(\Lambda'_\delta) \geq \alpha\mathsf{OPT}_\delta - \beta n$. Then Λ'_δ also satisfies $\Gamma(\Lambda'_\delta) \geq \alpha\mathsf{OPT} - (\beta + (\alpha+1)\epsilon)n$.*

We start with a random set of $n\rho = O(k \log n)$ initial nodes and fix them for the subsequent steps (Subsection 3.1). We then proceed to perform edge queries by probing their extended neighborhoods repeatedly (Subsection 3.2). In this way, we obtain $T = O(k \log n)$ cascades all starting from the same set of initial nodes (see Figure 4). 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 revealed connected component is large enough (Lemma 4). This is a key observation that allows us to upper-bound the total number of edge queries in Theorem 5. In Subsection 3.4, we propose the SEED algorithm to choose k seeds (approximately optimally), based on the outcome of the edge queries (the output of PROBE), and prove a $(1 - 1/e)\mathsf{OPT} - \epsilon n$ approximation guarantee with bounds on the number of edge queries (Subsection 3.4.1; Theorem 5) and the run time (Subsection 3.4.2; Theorem 7).

Recall from Section 2 and Definition 3 (edge queries) that each probing consists of independent random draws from the probed node's ordered neighborhood. Even if a node is probed more than once across different cascades, field survey researchers who devise their sampling plans based on the PROBE algorithm would only need to trace each revealed link once (having identified the unique links beforehand by recording T random draws from an ordered set). After the field survey is concluded, the data from all traced links can be collected to reconstruct the output of the PROBE algorithm based on the outcome of the random draws. In our analysis, we upper bound the total number of queries that the PROBE algorithm makes in all of the T cascades; therefore, the practical cost of tracing links during a field survey would be lower when the same edges appear in multiple cascades (e.g., the incident edge to the black node in Figure 4B is queried again in Figure 4C). In other applications — e.g., for a web crawler that follows the PROBE algorithm to mine data from online social networks (Catanese et al. 2011), bounding the total number of queried edges is a direct concern not only for scalability, but also to control the data collection costs and time.

3.1. Sampling the initial 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 $\lceil n\rho \rceil$ nodes uniformly at random. We call these the *initial nodes* and denote them by $\mathcal{V}_\rho$. Given $\mathcal{V}_\rho$, 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:

$$\Gamma_\rho(\mathcal{S}) := \frac{1}{\rho} \sum_{v \in \mathcal{V}_\rho} \phi(v, \mathcal{S}). \tag{2}$$

That is, we approximate the expected size of the cascade using the adoption probabilities of these initial nodes. 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 B.6. 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 2 (**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 extended neighborhoods of the initial nodes

Note that our definition of Γ_ρ in (2) is in terms of $\phi(v, \mathcal{S})$, which can only be computed given the knowledge of the entire graph. However, when access to graph information is restricted (network information is made available only through edge queries) we need to replace $\phi(v, \mathcal{S})$ by a proper estimate. To this end, we sample the graph edges through the probing procedure introduced in Section 2. Consider the $\lceil n\rho \rceil$ initial nodes in $\mathcal{V}_\rho$. 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 $\mathcal{G}_\rho^{(1)}, \ldots, \mathcal{G}_\rho^{(T)}$.

We can now estimate $\phi(v, \mathcal{S})$ for v belonging to $\mathcal{V}_\rho$ using the T subsampled graphs $\mathcal{G}_\rho^{(1)}, \ldots, \mathcal{G}_\rho^{(T)}$ as follows. For $i = 1, \ldots, T$, and $v \in \mathcal{V}_\rho$, 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 of $\phi(v, \mathcal{S})$ for $v \in \mathcal{V}_\rho$ and $\mathcal{S} \subset \mathcal{V}$ is

$$\phi^{(T)}(v, \mathcal{S}) := \frac{1}{T} \sum_{i=1}^{T} Y^{(i)}(v, \mathcal{S}). \tag{3}$$

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

$$\Gamma_\rho^{(T)}(\mathcal{S}) := \frac{1}{\rho} \sum_{v \in \mathcal{V}_\rho} \phi^{(T)}(v, \mathcal{S}). \tag{4}$$

To proceed, define

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

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 $v \in \mathcal{V}_\rho$ and $\mathcal{S} \subset \mathcal{V}$. The proof details are in Appendix B.7.

LEMMA 3 (**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 when we hit a threshold τ of 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 5). In fact, we show that it is safe to stop probing as soon as there are $\tau = \tau_\epsilon^{n,k}$ nodes in a connected component where $\tau_\epsilon^{n,k} := \lceil -(n \log \epsilon)/(\epsilon k) \rceil$.

Let us denote the T subsampled graphs obtained through limited probing by $\mathcal{G}_{\rho,\tau}^{(1)}, \ldots, \mathcal{G}_{\rho,\tau}^{(T)}$. Moreover, let $\Gamma_{\rho,\tau}^{(T)}$ be our estimate of the influence function that is constructed based on $\mathcal{G}_{\rho,\tau}^{(1)}, \ldots, \mathcal{G}_{\rho,\tau}^{(T)}$ in the exact same way as in (3) and (4). This new estimator is, itself, a submodular function since it can be expressed as a sum of coverage functions. Our following result ensures that by optimizing $\Gamma_{\rho,\tau}^{(T)}$ instead of $\Gamma_\rho^{(T)}$, we do not loose more than $(1-\epsilon)$ in our approximation factor. The proof follows a probabilistic argument similar to Bateni et al. (2017, Lemma 2.4). The crux of the argument is in constructing a random set whose expected value on $\Gamma_{\rho,\tau}^{(T)}$ 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 argue that any node whose connections are affected by limiting the probed neighborhoods should belong to a large component, of size $\tau = \tau_\epsilon^{n,k}$, and such large components are likely to be covered by one of the ϵk random nodes. The complete proof is in Appendix B.8.

LEMMA 4 (**Bounding the loss from limited probing**). *For $0 < \rho < 1$ and $0 < \epsilon \le 1$, 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 $\Gamma_{\rho,\tau}^{(T)}$ is an $\alpha(1-\epsilon)$-approximate solution to k-IM for $\Gamma_\rho^{(T)}$.*

Algorithm 2 summarizes the limited probing procedure for performing edge queries on the input graph $(\mathcal{G})$. The output is a sketch comprised of the T independent subsampled graphs $(\mathcal{G}_{\rho,\tau}^{(1)}, \ldots, \mathcal{G}_{\rho,\tau}^{(T)})$ that fully determine the estimator $\Gamma_{\rho,\tau}^{(T)}$.

3.4. Influence maximization on the sampled graph

Lemmas 2, 3, and 4 provide the following appropriate choices of the PROBE algorithm parameters ρ, T and τ:

$$\rho = \rho_{\epsilon,\delta}^{n,k} = \frac{(2+\epsilon)(\delta k \log n + \log 2)}{2\epsilon^2 n} \in O\left(\frac{\delta k \log n}{\epsilon^2 n}\right),$$

$$T = T_{\epsilon,\delta}^{n,k} = \left\lceil \frac{3(\delta + \log 2)(k+1) \log n}{\epsilon^2} \right\rceil \in O\left(\frac{\delta k \log n}{\epsilon^2}\right),$$

$$\tau = \tau_\epsilon^{n,k} = \left\lceil \frac{-n \log \epsilon}{\epsilon k} \right\rceil . \tag{5}$$

Algorithm 2: $\text{PROBE}(\rho, T, \tau, p)$

Input: Edge query access to graph $\mathcal{G}$, cascade probability p and probing parameters ρ, T and τ

Output: Subsampled graphs $\mathcal{G}^{(1)}_{\rho,\tau}, \ldots, \mathcal{G}^{(T)}_{\rho,\tau}$ and initial node set $\mathcal{V}_\rho$

1 Choose $\lceil n\rho \rceil$ nodes uniformly at random without replacement and call them $\mathcal{V}_\rho$.

2 for i **from** 1 **to** T **do**

3   Initialize $\mathcal{X} \leftarrow \varnothing$, $\mathcal{V}^{(i)}_{\rho,\tau} \leftarrow \mathcal{V}_\rho$, $\mathcal{E}^{(i)}_{\rho,\tau} \leftarrow \varnothing$ and $\mathcal{G}^{(i)}_{\rho,\tau} \leftarrow (\mathcal{V}^{(i)}_{\rho,\tau}, \mathcal{E}^{(i)}_{\rho,\tau})$.

      `// Construct` $\mathcal{G}^{(i)}_{\rho,\tau}$ `by probing the unexplored nodes (`$\mathcal{V}^{(i)}_{\rho,\tau} \setminus \mathcal{X}$`):`

4   **while** $\mathcal{V}^{(i)}_{\rho,\tau} \setminus \mathcal{X} \neq \varnothing$ **do**

5   Draw a node, ν, randomly from $\mathcal{V}^{(i)}_{\rho,\tau} \setminus \mathcal{X}$ add it to the explored nodes: $\mathcal{X} \leftarrow \mathcal{X} \cup \{\nu\}$.

6   Draw a random integer according to $\text{Binomial}\,(\text{card}(\mathcal{N}_\nu), p)$ distribution and call it I.

7   Draw a random subset of size I from $\{1, \ldots, \text{card}(\mathcal{N}_\nu)\}$ and call it $\mathcal{I}$.

8   **while** *size of the connected component of ν in $\mathcal{G}^{(i)}_{\rho,\tau}$ is less than τ* **do**

9   Draw an index, ι, randomly from $\mathcal{I}$ and remove it: $\mathcal{I} \leftarrow \mathcal{I} \setminus \{\iota\}$.

10   Perform a (ν, ι) edge query to graph $\mathcal{G}$ to reveal the ι-th neighbor of ν and call it ν_ι.

11   **if** $\nu_\iota \notin \mathcal{X}$ **then**

                `// Add the newly discovered node and edge to` $\mathcal{G}^{(i)}_{\rho,\tau}$`:`

12 $\mathcal{V}^{(i)}_{\rho,\tau} \leftarrow \mathcal{V}^{(i)}_{\rho,\tau} \cup \{\nu_\iota\}$

13 $\mathcal{E}^{(i)}_{\rho,\tau} \leftarrow \mathcal{E}^{(i)}_{\rho,\tau} \cup \{\nu, \nu_\iota\}$

14   **end**

15   **end**

16   **end**

17 $\mathcal{G}^{(i)}_{\rho,\tau} \leftarrow (\mathcal{V}^{(i)}_{\rho,\tau}, \mathcal{E}^{(i)}_{\rho,\tau})$.

18 end

19 return $\mathcal{G}^{(1)}_{\rho,\tau}, \ldots, \mathcal{G}^{(T)}_{\rho,\tau}$ *and* $\mathcal{V}_\rho$.

The following lemma (proved in Appendix B.9) combines our results so far (Lemmas 1 to 4) to show that with ρ, T and τ set according to (5) any α-approximate solution, $\Lambda^\star$, to k-IM on $\Gamma^{(T)}_{\rho,\tau}$ satisfies $\Gamma(\Lambda^\star) \geq \alpha' \text{OPT} - \epsilon' n$ for appropriate choices of α' and ϵ'; thus providing an approximate solution to the original k-IM problem on Γ.

LEMMA 5 (**Bounding the total approximation loss**). *Consider any $0 < \epsilon, \alpha < 1$, and fix $\rho = \rho^{n,k}_{\epsilon,\delta}$, $T = T^{n,k}_{\epsilon,\delta}$ and $\tau = \tau^{n,k}_\epsilon$ according to (5). 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 the k-IM problem on $\Gamma^{(T)}_{\rho,\tau}$ has value at least $\alpha' \text{OPT} - \epsilon' n$ on the original problem.*

In Subsection 3.4.1, we bound the total number of edges that are queried by $\mathrm{PROBE}(\rho, T, \tau, p)$. The output of the PROBE algorithm is the set of T subsampled graphs $(\mathcal{G}_{\rho,\tau}^{(1)}, \ldots, \mathcal{G}_{\rho,\tau}^{(T)})$. From these T subsampled graphs, we construct the estimator, $\Gamma_{\rho,\tau}^{(T)}$, and then use a submodular maximization algorithm to find a $(1 - 1/e - \epsilon)$-approximate solution to k-IM on $\Gamma_{\rho,\tau}^{(T)}$ 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 $O_\epsilon(pn^2 \log^4 n + \sqrt{kp}n^{1.5} \log^{5.5} n + kn \log^{3.5} n)$ time.

3.4.1. Bounding the total number of edge queries Our edge query upper bound includes the following terms:

$$
E_{\epsilon,p}^{n,k} := p\tau_\epsilon^{n,k}(\tau_\epsilon^{n,k} - 1)/2 \in O\left(\frac{pn^2 \log^2(1/\epsilon)}{\epsilon^2 k^2}\right), \tag{6}
$$

$$
C_{\epsilon,\delta}^{n,k} := n\rho_{\epsilon,\delta}^{n,k} T_{\epsilon,\delta}^{n,k} \left(1 + E_{\epsilon,p}^{n,k} + \sqrt{\delta(\tau_\epsilon^{n,k} \log n + \log T_{\epsilon,\delta}^{n,k}) E_{\epsilon,p}^{n,k}}\right)
$$

$$
\in O\left(\frac{\delta^2 \log^2(1/\epsilon)}{\epsilon^6} pn^2 \log^2 n + \frac{\delta^3 \log^{1.5}(1/\epsilon)}{\epsilon^{5.5}} \sqrt{kp}n^{1.5} \log^{2.5} n + \frac{\delta^2}{\epsilon^4} k^2 \log^2 n\right).
$$

In Theorem 5, we bound the total number of edge queries, denoted by q, in terms of $E_{\epsilon,p}^{n,k}$ and $C_{\epsilon,\delta}^{n,k}$. Our proof in Appendix B.10 relies critically on how we limit the probed neighborhoods (Subsection 3.3). Roughly speaking, the output of $\mathrm{PROBE}(\rho, T, \tau, p)$ consists of at most $n\rho T$ components of size no more than τ (barring the less than $n\rho T$ edges that may connect them). Moreover, since each edge 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 $\mathrm{PROBE}(\rho, T, \tau, p)$. 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 steps 10 and 11 of the PROBE algorithm).

THEOREM 5 (**Bounding the edge queries**). *Consider any $0 < \epsilon, \alpha < 1$, fix $\rho = \rho_{\epsilon,\delta}^{n,k}$, $T = T_{\epsilon,\delta}^{n,k}$ and $\tau = \tau_\epsilon^{n,k}$ according to (5), and denote the total number of edge queries during a single run of $\mathrm{PROBE}(\rho, T, \tau, p)$ by q. For $n \geq \sqrt{\delta + \log T}$ with probability at least $1 - 3e^{-\delta}$, q can be bounded as follows:*

$$
q \leq Q_{\epsilon,\delta}^{n,k} := 2C_{\epsilon,\delta}^{n,k} + \left(2 + \sqrt{2}\right) T_{\epsilon,\delta}^{n,k} n \sqrt{\delta + \log T_{\epsilon,\delta}^{n,k}}
$$

$$
\in O\left(\frac{\delta^2 \log^2(1/\epsilon)}{\epsilon^6} pn^2 \log^2 n + \frac{\delta^3 \log^{1.5}(1/\epsilon)}{\epsilon^{5.5}} \sqrt{kp}n^{1.5} \log^{2.5} n + \frac{\delta^2}{\epsilon^4} k^2 \log^2 n\right.
$$

$$
\left. + \frac{\delta^{1.5} \log^{0.5}(1/\epsilon)}{\epsilon^2} k\sqrt{\log k}n \log n \sqrt{\log \log n}\right)
$$

$$
\subset O_{\epsilon,\delta}\left(pn^2 \log^2 n + \sqrt{kp}n^{1.5} \log^{2.5} n + kn \log^2 n\right). \tag{7}
$$

It is worth highlighting that to provide our main approximation guarantee in Theorem 7, we set $\delta = 2\log n$; see Appendix B.12. In Appendix B.11, we prove a matching (up to logarithmic factor) lower bound by giving a hard example where it is impossible to provide a $\mu\mathsf{OPT} - \epsilon n$ approximate guarantee using $o(pn^2)$ edge queries. We allow p to vary with n and prove the hard case for $k = 1$ and $p \in \Omega(\log n/n)$, whereby the first term in (7) is dominant and we have $Q_{\epsilon,\delta}^{n,k} \in O_{\epsilon,\delta}(pn^2 \log^2 n)$; of note, replacing $p \in O(\log n/n)$ in (7) yields $Q_{\epsilon,\delta}^{n,k} \in O_{\epsilon,\delta}(kn \log^3 n)$. Our hard example builds on our previous construction in Figure 5, which is a variant of the so-called "caveman graph" where edges are rewired to link different cliques. In Appendix B.11, we provide a construction consisting of $9/\mu^2$ cliques, $3/\mu$ of which are connected around a circle, by choosing $-p^{-1}\log(\gamma\mu/6)$ edges randomly from each clique and rewiring them to connect to the next clique on the circle, while preserving the node degrees. Here $0 < \gamma < \mu/6$ is a constant that is chosen arbitrarily and then fixed. With $k = 1$, an optimal algorithm seeds one of the nodes in the $3/\mu$ connected cliques and achieves an expected spread size of at least $n(1-\gamma)\mu/3$. To show that no approximation algorithm can provide a $\mu\mathsf{OPT} - \epsilon n$ guarantee using $o(pn^2)$ edge queries, we consider an arbitrary algorithm that makes less than $pC_{\mu,\gamma}^{\epsilon}n^2$ edge queries, where $C_{\mu,\gamma}^{\epsilon}$ is a constant that is specified in Appendix B.11. The probability that such an algorithm detects the $3/\mu$ connected cliques is at most $\mu/3 - 3\epsilon/\mu$. Hence, the expected spread size from seeding the output of such an algorithm cannot exceed $n(1-\gamma)\mu^2/3 - \epsilon n$ which is strictly less than $\mu\mathsf{OPT} - \epsilon n$.

THEOREM 6. *Let $0 < \mu \le 1$ be any constant. There can be no approximation algorithms that provide $\mu\mathsf{OPT} - \epsilon n$ guarantees for k-IM using $o(pn^2)$ edge queries when $\epsilon < \mu^2/18$.*

Of note, the upper bound ($\mu^2/18$) on ϵ in Theorem 6 is necessary for limiting the probability of querying a rewired edge from the connected cliques. An upper bound of $\mu/3 - 3\epsilon/\mu > 0$ on this probability implies an overall $\mu\mathsf{OPT} - \epsilon n$ upper bound on the performance of any algorithm on this hard input. It seems unavoidable that this hardness should hold only for small ϵ relative to μ, however, the exact quadratic dependence on μ may be improvable with a significantly different construction.

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 $\Gamma_{\rho,\tau}^{(T)}$ is, by definition, a coverage function, ergo a submodular function. Hence, we can use the randomized greedy algorithm of Mirzasoleiman et al. (2015) to provide a $(1 - 1/e - \epsilon)$ approximation guarantee. We start with $\Lambda^\star \leftarrow \varnothing$ 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 $\Lambda^\star$, denoted by:

$$\Delta(v|\Lambda^\star) := \Gamma_{\rho,\tau}^{(T)}(\Lambda^\star \cup \{v\}) - \Gamma_{\rho,\tau}^{(T)}(\Lambda^\star).$$

- We choose a node $v^\star$ with maximal marginal increase and add it to the seed set:

  $\Lambda^\star \leftarrow \Lambda^\star \cup \{v^\star\}$.

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

Algorithm 3: $\mathrm{SEED}(\epsilon, k)$

Input: Subsampled graphs $\mathcal{G}^{(1)}_{\rho,\tau}, \ldots, \mathcal{G}^{(T)}_{\rho,\tau}$ and initial node set $\mathcal{V}_\rho$

Output: $\Lambda^\star$, $(1 - 1/e - \epsilon)$-approximate solution to k-IM for $\Gamma^T_{\rho,\tau}$

1 Initialize $\Lambda^\star \leftarrow \varnothing$.

2 Find the connected components of $\mathcal{G}^{(1)}_{\rho,\tau}, \ldots, \mathcal{G}^{(T)}_{\rho,\tau}$.

3 Initialize the value of every connected component to be equal to the number of its nodes in
  $\mathcal{V}_\rho$.

4 for i **from** 1 **to** k **do**
  // Selecting the i-th seed following a randomized greedy step:

5   Choose a subset $\mathcal{R}$ from $\mathcal{V} \setminus \Lambda^\star$ randomly with $\mathrm{card}(\mathcal{R}) = (n/k)\log(1/\epsilon)$.

6   **for** $v \in \mathcal{R}$ **do**
7   Set $\Delta(v|\Lambda^\star)$ equal to the sum of the values of the connected components containing v.

8   **end**

9 $v^\star \leftarrow \arg\max_{v \in \mathcal{R}} \Delta(v|\Lambda^\star)$.

10 $\Lambda^\star \leftarrow \Lambda^\star \cup \{v^\star\}$.

11   Update the values of the connected components containing $v^\star$ to zero.

12 end

13 return $\Lambda^\star$.

In Algorithm 3: $\mathrm{SEED}(\epsilon, k)$, we provide efficient implementations for the above operations. Our implementations are based on the structure of $\Gamma^{(T)}_{\rho,\tau}$, as determined by the T subsampled graphs $(\mathcal{G}^{(1)}_{\rho,\tau}, \ldots, \mathcal{G}^{(T)}_{\rho,\tau})$. 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 $\mathcal{V}_\rho$) 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 $\Gamma^{(T)}_{\rho,\tau}$ is equivalent to finding a seed set, $\Lambda^\star$, such that the total value of all connected components containing at least one seed in $\Lambda^\star$ is maximized. If a connected component already contains (i.e., is covered by) some nodes in $\Lambda^\star$, 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 seed from that component to $\Lambda^\star$ — see step 11 of Algorithm 3.

In Algorithm 4 we run PROBE and SEED under the right parameter setting to achieve a $(1 - 1/e)\mathsf{OPT} - \epsilon'n$ approximation guarantee for influence maximization. Theorem 7 formalizes our guarantees by combining our conclusions from Lemma 5 and Theorem 5, as well as the analysis of the performance of fast submodular maximization (randomized greedy) in Mirzasoleiman et al. (2015). The proof is in Appendix B.12.

Algorithm 4: k-IM with a bounded number of edge queries

Input: Edge query access to graph $\mathcal{G}$ on n nodes and an approximation loss $\epsilon' > 0$

Output: An approximate k-IM solution, $\Lambda^\star$, satisfying $\Gamma(\Lambda^\star) \geq (1 - 1/e)\mathsf{OPT} - \epsilon'n$

// Setting the parameters $(\epsilon, \delta, \rho, T, \tau)$:

1 $\epsilon \leftarrow \epsilon'/7$·

2 $\delta \leftarrow 2\log n$·

3 $\rho \leftarrow (2 + \epsilon)(\delta k \log n + \log 2)/(2\epsilon^2 n)$·

4 $T \leftarrow \lceil 3(\delta + \log 2)(k + 1)\log n/\epsilon^2 \rceil$·

5 $\tau \leftarrow \lceil n\log(1/\epsilon)/(\epsilon k) \rceil$·

// Running PROBE followed by SEED:

6 Run $\mathrm{PROBE}(\rho, T, \tau, p)$ with edge query access to graph $\mathcal{G}$ to obtain $\mathcal{G}^{(1)}_{\rho,\tau}, \ldots, \mathcal{G}^{(T)}_{\rho,\tau}$ and $\mathcal{V}_\rho$.

7 Run $\mathrm{SEED}(\epsilon, k)$ with $\mathcal{G}^{(1)}_{\rho,\tau}, \ldots, \mathcal{G}^{(T)}_{\rho,\tau}$ and $\mathcal{V}_\rho$ as inputs to obtain $\Lambda^\star$.

8 return $\Lambda^\star$.

THEOREM 7. *For any $0 < \epsilon' \leq 1$ and $n \geq (30/\epsilon')^2$, there exists an algorithm for influence maximization that covers $(1 - 1/e)\mathsf{OPT} - \epsilon'n$ nodes in expectation with $O_{\epsilon'}(pn^2\log^4 n + \sqrt{kp}n^{1.5}\log^{5.5} n + kn\log^{3.5} n)$ expected run time and using no more than $O_{\epsilon'}(pn^2\log^4 n + \sqrt{kp}n^{1.5}\log^{5.5} n + kn\log^{3.5} n)$ edge queries in expectation; the dependence of the constant factors on ϵ' is the same as (7) with $\epsilon = \epsilon'/7$.*

3.5. Discrepancy between the query and cascade probabilities

So far we have assumed that the cascade probability (p) is known or otherwise available to perform $\mathrm{PROBE}(\rho, T, \tau, p)$ in step 6 of Algorithm 4. While p can be measured beforehand, in practice, such measurements are subject to significant uncertainty. Even when the network is fully observed, there can be uncertainty about p (Chen et al. 2016, He and Kempe 2016); however, here p plays a role in data collection, so it is important to consider whether other values of p can be entertained after data collection. Motivated by the possibility that p may be unknown and our best estimate of p can change after the fact (i.e., after data collection), let us assume that the edge data is

collected according to $\text{PROBE}(\rho, T, \tau, p')$ where the "probe probability" (p') is different from the target cascade probability (p). Note that the setting of PROBE parameters $(\rho, T$ and $\tau)$ in steps 1 to 5 of Algorithm 4 is not dependent on p'. Let $\mathcal{G}'^{(1)}_{\rho,\tau}, \ldots, \mathcal{G}'^{(T)}_{\rho,\tau}$ be the subsampled graphs output by $\text{PROBE}(\rho, T, \tau, p')$. The following hardness result shows that when p is unknown, controlling the discrepancy between the query and seed cascade probabilities by a multiplicative ratio (i.e., requiring $|\max\{p, p'\}/\min\{p, p'\}| < 1 + \delta$) is not enough for providing general $\mu\mathsf{OPT} - \epsilon n$ approximation guarantees:

THEOREM 8. *Fix $0 < \mu < 1$ and $0 < \delta < 1$. Let p' and p be the independent cascade probabilities for probing and seeding. There can be no approximation algorithms that provide $\mu\mathsf{OPT} - \epsilon n$ guarantees for k-IM when the querying and seeding cascade probability are different, $p \neq p'$, with $\max\{p, p'\}/\min\{p, p'\} = 1 + \delta$, $c_\delta = 1 + (\delta - \delta^2)/2$ and*

$$\epsilon < \frac{8\mu(\delta - \delta^2)c_\delta}{41(2-\mu) + 16(\delta - \delta^2)c_\delta}.$$

Stability and robustness of influence maximization are well-studied in light of uncertain cascade probabilities (Chen et al. 2016, He and Kempe 2016, 2018, Wilder et al. 2017b). Chen et al. (2016, Theorem 3) use phase transition behavior in Erdős-Rényi graphs to show that an $O(1/n)$ additive perturbation is enough to reduce the robust ratio (defined as the maximum over all seed sets of the minimum of approximation ratio of a seed set over the parameter range) to $O(\log n/n)$. In Appendix B.13, we use a similar construction to prove Theorem 8. Let us denote $\hat{p} = \max\{p, p'\}$ and $\check{p} = \min\{p, p'\}$. Our hard example consists of a clique, $\mathcal{G}_1$, of size $2\epsilon n/\mu$, and a second subgraph, $\mathcal{G}_2$, that is obtained from the realization of a random graph with edge probability $(1 - \delta/2)/(n_{\epsilon,\mu}\check{p})$ on the remaining $n_{\epsilon,\mu} := (1 - 2\epsilon/\mu)n$ nodes. We choose $\check{p} = \Omega(\log n/n)$ such that active edges on $\mathcal{G}_1$ constitute a connected component of size of ϵn, with high probability as $n \to \infty$. However, when the cascade probability is $\check{p}$, the active edges on $\mathcal{G}_2$ constitute a random graph with edge probability $(1 - \delta/2)/n_{\epsilon,\mu}$ so that the the largest connected component on $\mathcal{G}_2$ is with high probability $O(\log n)$. For $k = 1$, it is optimal to choose one of the $2\epsilon n/\mu$ nodes in $\mathcal{G}_1$ as the seed when the cascade probability is $\check{p}$. On the other hand, if the cascade probability is $\hat{p} = (1 + \delta)\check{p}$, then the active edges induce a random graph with edge probability $c_\delta/n_{\epsilon,\mu}$ on $\mathcal{G}_2$ where $c_\delta = 1 + (\delta - \delta^2)/2 > 1$. With high probability as $n \to \infty$, this random graph contains a giant connected component of size $f_\delta n$, satisfying $f_\delta = 1 - e^{-f_\delta c_\delta} > 1 - e^{-c_\delta} > 2\epsilon n/\mu$. In Appendix B.13, we use common techniques from connectivity analysis of random graphs and the emergence of the giant connected component therein to bound the expected spread size from seeding a node in either of the two subgraphs ($\mathcal{G}_1$ and $\mathcal{G}_2$). Subsequently, we show that for ϵ sufficiently small, any k-IM approximation algorithm that provides a $\mu\mathsf{OPT} - \epsilon n$ guarantee, should necessarily seed $\mathcal{G}_1$ when the cascade probability is $\check{p}$ and

$\mathcal{G}_2$ when the cascade probability is $\hat{p}$. Therefore, no such approximation algorithm exists when when the probe and cascade probabilities are different, i.e., with $\hat{p} \neq \check{p} = 1 + \delta > 1$.

If p is known and the PROBE(ρ, T, τ, p') data is collected such that $p' > p$, then one can use a simple pruning procedure to correct for the difference between p' and p by removing the edges of $\mathcal{G}'^{(1)}_{\rho,\tau}$, ..., $\mathcal{G}'^{(T)}_{\rho,\tau}$ independently with probability $1 - p/p'$. Given the output of PROBE(ρ, T, τ, p'), the algorithm in Appendix B.14 performs such a pruning and returns a corrected set $\mathcal{G}^{(1)}_{\rho,\tau}, \ldots, \mathcal{G}^{(T)}_{\rho,\tau}$ that exactly simulates the output of PROBE(ρ, T, τ, p). On the other hand, the dependent sampling process by which $\mathcal{G}'^{(1)}_{\rho,\tau}$, ..., $\mathcal{G}'^{(T)}_{\rho,\tau}$ are generated prevents us from using a similar post-processing (e.g., by combining $\mathcal{G}'^{(1)}_{\rho,\tau}$, ..., $\mathcal{G}'^{(T)}_{\rho,\tau}$ into union graphs) when $p' < p$. The reason is that different edges have different probability of appearing in a subsampled graph $(\mathcal{G}'^{(i)}_{\rho,\tau})$, depending on their network location and realization of other edges in $\mathcal{G}'^{(i)}_{\rho,\tau}$ — edges that are connected to more influential nodes or other queried edges are more likely to be queried. In practical applications of edge queries (e.g., when running surveys to help diffuse health interventions), one can incentivize survey participants to reveal more edges and trace enough of them to ensure $p' > p$, albeit at an increased cost.

4. Costs and benefits of network information

We can study the value of network information by examining how the expected spread size changes as more queries are used to select the seeds; that is, we can vary T and ρ in our algorithms. In this section, simulations of spread sizes with increased queries on an empirical network indicate the existence of an inflection point, whereby the first few queries improve the performance significantly before hitting a notably diminished returns. When this is the case, we can extract the benefits of the network information using just a few queries.

In particular, we conduct simulations with the Pennsylvania State University (Penn State) Facebook social network, with $41,536$ nodes, average degree 65.59, and a total of $1,362,220$ edges. It is the largest network in a collection of Facebook social networks in 100 U.S. colleges and universities described by Traud et al. (2012). Note that although the social network is known by the platform in advance, the friends lists are not available to, e.g., the electronic commerce companies that operate on the Facebook platform and can be collected through costly effort.

Figure 6 shows the performance of Algorithms 2 and 3 (PROBE & SEED) on the Penn State Facebook social network. Running the PROBE algorithm with higher values of T leads to discovery of more nodes and edges from the social network. The vertical axes show the mean spread sizes from seeding the output of Algorithm 3 for each T using 50 random inputs. Recall that each input is a set of T probed samples $\mathcal{G}^{(1)}_{\rho,\tau}, \ldots, \mathcal{G}^{(T)}_{\rho,\tau}$ that is obtained through Algorithm 2. The output performance improves with increasing T, since with more nodes and edges revealed, the output seed set can be better optimized; nevertheless, there are diminishing returns to the increasing network information.
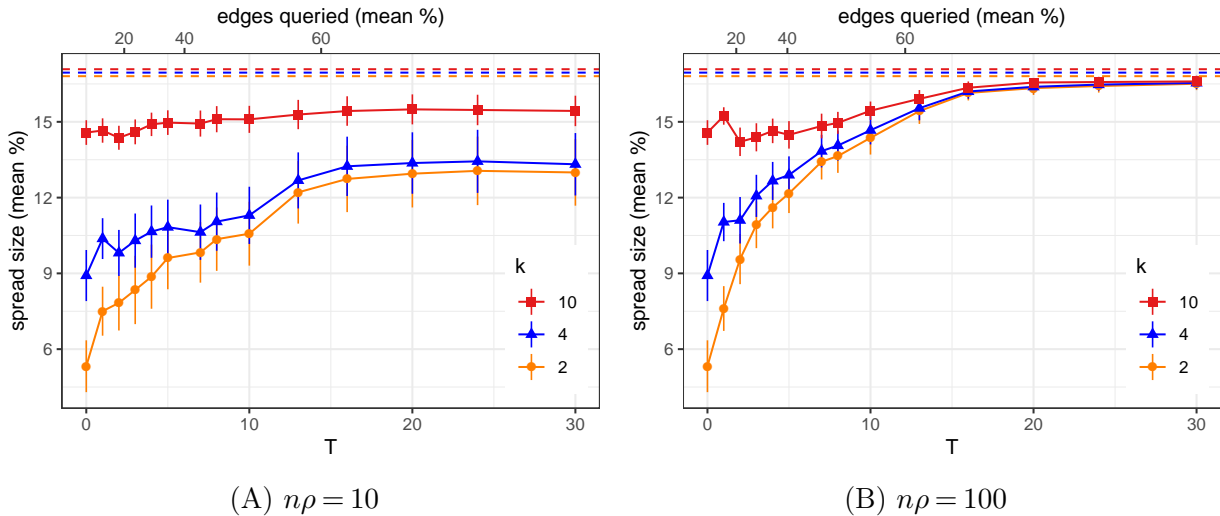
(A) $n\rho = 10$ (B) $n\rho = 100$

Figure 6 The mean spread sizes from seeding the output of Algorithm 3 applied to the Penn State Facebook social network as T is increased for (A) 10 and (B) 100 initial nodes. 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 subsampled graphs $(\mathcal{G}_{\rho,\tau}^{(1)}, \ldots, \mathcal{G}_{\rho,\tau}^{(T)})$ that are input to the SEED algorithm, we run the PROBE algorithm starting from $n\rho$ initial nodes, randomly chosen, and vary T over a logarithmic scale: $T \in \{0, 1, 2, 3, 4, 5, 7, 8, 10, 13, 16, 20, 24, 30\}$. 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 the SEED algorithm. The vertical axes in (A) and (B) show the mean spread sizes and 95% confidence intervals that are computed over the 50 outputs of the SEED algorithm for each T. Their top axes show the average number of revealed edges that is computed over the 50 random inputs for each T. The complete-information, greedy baseline at each k is marked by a dashed line.

Figure 6B shows that we can extract the benefits of complete network information using just $T = 30$ iterations: 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.

It is worth noting that the random variations in the algorithm output — hence, the width of the confidence intervals — also decreases with the increasing network information in the input. There are two sources of randomness in the SEED algorithm's performance: $T < \infty$ 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 $(\mathrm{card}(\mathcal{V}_\rho) = n\rho)$, from 10 nodes in Figure 6A to 100 nodes in Figure 6B allows us to remove the remnant randomness from the algorithm output at large T.

Our algorithms and these simulation results add important nuance to recent discussions of the value of network information. Akbarpour et al. (2020) show that, for special classes of random graph inputs, seeding $k + x$ nodes at random (using no information about the network) for some $x \in \omega(1)$ is enough to outperform the optimum spread size with k nodes, as the network size increases $(n \to \infty)$.

(A) $C_S = 100$, $C_T = 200$
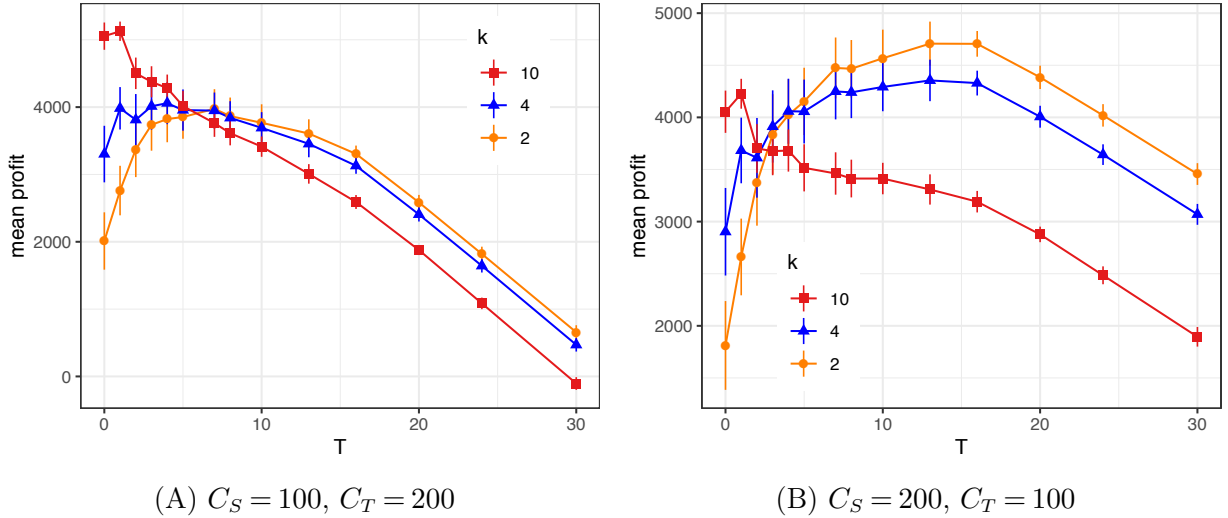
(B) $C_S = 200$, $C_T = 100$

Figure 7   The mean profit from seeding the output of Algorithm 3, given the cost of seeds (C_S per seed), iterations cost (C_T per iteration), and unit revenue per adopter. The vertical axis shows the mean profits and confidence intervals that are computed from 50 executions of the algorithms with increasing T and 100 initial nodes. (7A) When $C_S = 100$ and $C_T = 200$, the maximum expected profit is achieved at $k = 10$ with no queries. (7B) Increasing the cost of seeds to $C_S = 200$ and decreasing the cost of iterations to $C_T = 100$ changes the optimal operating point to $k = 2$ seeds with $T = 13$ iterations.

They conclude that the benefits of acquiring network information to identify the optimal k seeds can be offset by seeding a few more nodes at random (without using any network information). We complement the results of Akbarpour et al. (2020) in two ways. First, we measure how many queries are needed to yield the same expected spread sizes achievable using full knowledge of the network. Second, in our framework we can make the trade-off between acquiring network information and using more seeds explicit by seeding more nodes and reducing the number of queries to keep the performance fixed. If we assume a cost, C_S, for each seeded node and another cost, C_T, for running each PROBE iteration, and a unit revenue per each adoption, then there is a number of iterations that is expected profit maximizing for a given number of seeds. In Figure 7, the maximum expected profit for $C_S = 100$ and $C_T = 200$ is achieved with $k = 10$ and $T = 0$ iterations, i.e., randomly seeding with the largest seed set size considered (Figure 7A). However, increasing the cost of the seeds to $C_S = 200$ and decreasing the cost of the iterations to $C_T = 100$ reverses this result (Figure 7B), and the optimum operating point shifts to $k = 2$ and $T = 13$.

5. Discussion and future directions

We addressed the problem of choosing the k most influential nodes when the social network is unknown and accessing it is costly. We analyzed two ways of acquiring network information by observing influence spread of random nodes (influencing sampling) and by revealing the identity of

neighboring nodes in an adjacency list (edge queries). We provided polynomial-time algorithms with almost tight approximation guarantees using a bounded number of queries to the graph structure (Theorems 2 and 7). We also provided hardness results to show that multiplicative approximation guarantees are generally impossible (Theorems 1 and 4) and to lower-bound the query complexity and show tightness of our query bounds for providing approximation guarantees with an additive loss (Theorems 3 and 6). Finally, we showed the utility of our bounded-query framework for studying the trade-off between the cost of acquiring more network information and the benefit of increasing the spread size.

The preceding results were for the independent cascade model over undirected graphs with a homogeneous cascade probability. In Appendix C.1, we discuss the extension of our results to directed graphs. In Appendix C.2, we explain how our techniques can be applied to other commonly posited models of diffusion, and in particular, we prove the following extension for the linear threshold model (Kempe et al. 2015):

THEOREM 9. *For any arbitrary $0 < \epsilon \le 1$, there exists a polynomial-time algorithm for influence maximization under the linear threshold model that covers $(1 - 1/e)\mathsf{OPT} - \epsilon n$ nodes in expectation in $O(nk^2 \log(n/\epsilon)/\epsilon^3)$ time, using no more than $nk\lceil 81k\log(6nk/\epsilon)/\epsilon^3\rceil \in O(nk^2 \log(n/\epsilon)/\epsilon^3)$ edge queries.*

Results that address problem of seeding with partial network information are nascent and we foresee many directions for future research in this area. It is possible to provide tighter approximation guarantees or better query bounds if the input graph follows a known distribution, e.g., the stochastic block model (Wilder et al. 2018) or if individuals can directly report on who is influential by, e.g., recalling frequent origins of past cascades (Banerjee et al. 2019, Flodgren et al. 2011). Thus, an important venue for future work is to explore other ways of acquiring information about the graph structure. For example, one can draw inspiration from the graph sampling literature to devise new query methods (Leskovec and Faloutsos 2006) to obtain subsampled graphs that preserve enough network information to perform influence maximization satisfactorily. We are particularly interested in queries that measure the spread of influence subject to time constraints. This is especially relevant in practice when spending time on data collection is costly and decision-makers have preference for earlier rather than later adoptions (Libai et al. 2013) or prefer diffusions among certain subgroups more than others. We speculate that operational considerations such as unequal, time-critical adoptions and privacy concerns in data collection open new venues for future methodological and applied works that build on the same foundation as ours.

Broadly, our results highlight the importance of thinking through data collection in conjunction with planned interventions. Natural sampling methods can be re-designed to optimize for intervention outcomes (Algorithms 1 and 2). Beyond the co-design of sampling and targeting algorithms

presented here, it is important to plan the data collection efforts with attention to their intervention contexts. For example, in the absence of reliable information about the spread, collecting network data without attention to diffusion parameters can lead to unsatisfactory outcomes (Theorem 8). Our lower bounds (Theorems 3 and 6) point to the implied cost of data collection, which in practice can pose a major bottleneck. Notwithstanding, explicit understanding of these costs helps practitioners in value of information analysis and in deciding on how to allocate their limited resources between data collection and increased intervention (Section 4). Future work can bring out other trade-offs that are inherent in this co-design framework, e.g., by focusing on the privacy costs of data collection and its benefits to various subgroups (measured in terms of the intervention outcomes). With the increasing prevalence of data-driven intervention designs and policy practices, understanding such trade-offs has important implications for social welfare.

Acknowledgments

Appendix A: Additional related work

Our INF-SAMPLE algorithm uses $O(k^2 \log n)$ influence samples to seed k nodes approximately optimally (Theorem 2). Our $(1-1/e)\mathsf{OPT} - \epsilon n$ approximation guarantee for INF-SAMPLE matches the $(1-1/e)\mathsf{OPT}$ multiplicative factor that is tight for k-IM but suffers an ϵn additive loss. Theorem 1 shows that the additive loss is impossible to avoid with a sub-linear sample size. Theorem 3 shows that even an ϵn additive guarantee is impossible if the number of samples is sub-quadratic in k, therefore, the quadratic order dependence of sample size on k is also hard to improve. In the PROBE algorithm, we organize our edge queries to approximately simulate $T \in O(k \log n)$ independent cascades each starting from the same set of $n\rho \in O(k \log n)$ randomly sampled nodes. The nuanced implementation of PROBE allows us to construct these simulations using only a bounded number of edge queries, while still preserving enough information in its output to select k seeds approximately optimally by applying the SEED algorithm (Theorem 7).

Since an earlier version of the present work (Eckles et al. 2019), sample complexity of influence maximization is treated thoroughly by Sadeh et al. (2020). Through a careful analysis, Sadeh et al. (2020) are able to upper-bound the variance of the number of adopters at a fixed diffusion step in term of the expected number of adopters and the diffusion step. This variance upper bound allows them to efficiently control the estimation error of the spread sizes by relaxing the relative error requirements when seed sets produce small spreads. Subsequently, Sadeh et al. (2020) give a sample complexity bound of $O(kt \log n)$ where t is the number of diffusion steps. This is the best known upper bound on the number of simulations required for achieving tight k-IM approximation guarantees but is still super-linear in the worst-case because the number of diffusion steps t can be of the order of the network size n. It is worth noting that the $O(kt \log n)$ bound applies to i.i.d. simulations of the spread and is achieved using an adaptive sample size — their implementation of the approximate greedy algorithm uses $O(k^3 t \log n)$ simulations, and they are not directly concerned with limiting access to the input graph. On a broader scope, reverse influence sampling, i.e., collecting influence samples on the transposed graph with edge directions reversed — proposed by Borgs et al. (2014) — has been successfully applied in the design of k-IM algorithms with practical efficiency and near-optimal run time (Nguyen et al. 2016, Tang et al. 2015, 2014). For undirected graphs we can collect our influence samples on the original graph (as opposed to its transpose), and by allowing for an ϵn additive loss, we can achieve our $(1-1/e)\mathsf{OPT} - \epsilon n$ guarantee on general graphs using $O(k^2 \log n)$ influence samples; thus avoiding the linear dependence on n which is prohibitive when network information is limited and costly.

More broadly, the influence sampling framework relates to recent advances in application of stochastic oracles for submodular maximization with imperfect information. Karimi et al. (2017)

consider a class of discrete optimization problems where the objective function is expressed as an expectation over submodular functions and can be estimated by sample averaging but is not explicitly available and cannot be used as a black box oracle for the greedy algorithm. Their approach is by lifting the objective function into the continuous domain using a concave upper-bound on its multilinear extension. The concave upper-bound is guaranteed to be no more than $e/(e-1)$-times the optimization objective and can be maximized efficiently through projected stochastic gradient ascent. The finally transfer the solution back to the discrete domain using a randomized rounding technique that preservers the quality of approximation in expectation. The fast convergence result of Karimi et al. (2017) applies to the total number of gradient steps required for maximizing the concave relaxation of the objective function and is given by $T = \lceil B^2 \rho^2 / \epsilon^2 \rceil$ where ϵ is an additive loss in the approximation guarantee, ρ is a bound on the gradient norms, and B is a bound on the norm of the continuous optimization variable (Karimi et al. 2017, Theorem 2). In the case of k-IM on an n node network while subgradients can be estimated by BFS on influence samples, with $\rho = \sqrt{n}$ and $B = \sqrt{k}$, we need a total of $T = O(nk)$ stochastic gradient steps, thus suffering the same prohibitive linear dependence on n (Karimi et al. 2017, Lemma 4).

Another related body of literature studies optimizing submodular functions based on input-output data pairs that are sampled from a distribution over feasible inputs (e.g., uniformly over all input sets of size k). In the k-IM setup, this learning-theoretic framework implies that the observed data consist of pairs of initial nodes and expected number of adopters for cascades initiated from those nodes. Balkanski et al. (2017b) show hardness of approximation for maximizing coverage functions under cardinality constraints (including k-IM) using polynomially many samples from any distribution, whereas for monotone submodular functions with bounded curvature more positive results can be achieved with polynomial sample size (Balkanski et al. 2016). Our influence sampling framework is principally different from the learning-theoretic framework of optimization from samples because we cannot observe the exact value of the influence function but instead see a random realization of the adopters for each input. Moreover, each influence sample starts from a random initial node and our inputs are not constrained to be subsets of size k. Balkanski et al. (2017a) consider an adaptation of the optimization from samples framework to k-IM where each sample consists of the number of adopters in a random cascade. They offer an algorithm to list nodes in the order of their decreasing expected marginal contributions to a random set, and then iteratively remove those whose marginal contribution significantly overlaps with an earlier node on the list. Estimating marginal contributions in this framework requires polynomially many samples (Balkanski et al. 2017a, Lemma 15; the exact dependence on n is not clarified but $O(n)$ appears to suffice), while the performance guarantees are applicable only to stochastic block model random input graphs, (Balkanski et al. 2017a, Theorems 6 and 12).

Appendix B: Proofs & other mathematical details

B.1. Proof of Theorem 1: Hardness of approximation with influence samples

Pick an arbitrary function $f(n) \in o(n)$, and let $g(n) = \sqrt{n/f(n)}$. Note that $g(n) \in \omega(1)$. Consider an algorithm Alg that uses $f(n)$ influence samples. Whe show that Alg cannot be a μ-approximation. Our hard example consists of a clique of size $g(n)$, chosen uniformly at random, and $n - g(n)$ isolated nodes and we aim to seed one node (Figure 1). One can bound the probability 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 see a node via influence samples, it seeds one of the nodes of the clique with probability at most $\frac{g(n)}{n - f(n)} \in o(1)$. 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 $\frac{o(1)g(n)+1}{g(n)} \in o(1)$ as claimed.

B.2. Proof of Theorem 2: Approximation guarantees with bounded influence samples

Recall our notation in the INF-SAMPLE algorithm. The output of the algorithm, $\Lambda^\star$, is a set of k nodes that are chosen, one by one, in k iterations. Let us use $\Lambda^{\star i}$ to denote the first selected i seeds. In the i-th iteration, we choose ρ initial nodes at random (with replacement) and collect influence samples. We use A_j^i to denote the j-th influence sample collected during the i-th iteration. We reset A_j^i to $\varnothing$ if it contains any of the $i - 1$ nodes selected in the previous iterations. 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 A_j^i's. 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, ..., A_ρ^i. Subsequently, in step i, we choose $v^\star = \arg\max_{u \in \mathcal{V} \setminus \Lambda^{\star i-1}} X_u^i$ and add it to $\Lambda^\star$.

We analyze the iterations of Algorithm 1 and show that for $\epsilon' = \epsilon/3$ and $\rho = \rho_\epsilon^{n,k} = \lceil 3k \log(2nk/\epsilon')/\epsilon'^3 \rceil$, the output of INF-SAMPLE(ρ, k) satisfies the desired approximation guarantee. Let us define random variable N_u^i to be the expected number of nodes that are covered by $\Lambda^{\star i-1} \cup \{u\}$ but not by $\Lambda^{\star i-1}$. Note 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 $v^\star \in \mathcal{V} \setminus \Lambda^{\star i-1}$ to maximize $\mathbb{E}[N_u^i]$ is equivalent to one step of the greedy algorithm. This is equivalent to choosing $v^\star \in \mathcal{V} \setminus \Lambda^{\star i-1}$ 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 at the i-th iteration satisfying $\mathbb{E}[N_u^i] < \epsilon'n/k$ for all candidates $u \in \mathcal{V} \setminus \Lambda^{\star i-1}$, then 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 when $\mathbb{E}[N_u^i] < \epsilon'n/k$ for all u, i.e., the algorithm stops if it selects k seeds or

$\mathbb{E}[N_u^i] < \epsilon' n/k$ for all u, whichever comes first. This allows us to lower-bound the expected spread size from seeding the output of Algorithm 1. In reality, any additional node that the algorithm selects will only improve the expected spread size of its output. 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

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

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 an algorithm is called $(1-2\epsilon')$-approximate greedy in Golovin and Krause (2011) and it is proven to return a $(1-1/e-2\epsilon')$-approximate solution Badanidiyuru and Vondrák (2014), Golovin and Krause (2011), Kumar et al. (2015). Therefore, we can bound the expected value of the output solution of INF-SAMPLE(ρ, k) as follows:

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

B.3. Proof of Theorem 3: Lower-bounding the required number of influence samples

Fix any $0 < \mu < 1$, $0 < \epsilon < \mu/k$, and set $\beta = 0.5(\mu - k\epsilon - 1 - 2/k)$. Consider our hard example in Figure 3 with $n > k^2$. Note that given the complete network information in this example, the optimum strategy is to seed the k centers of each of the k stars, which achieves $\mathsf{OPT} = k(n/k^2) = n/k$ expected spread size. Let Alg be any algorithm that seeds optimally using less than βk^2 influence samples. We will show that the expected spread size from seeding the output of Alg is strictly less than $\mu\mathsf{OPT} - \epsilon n$. To upper-bound the expected spread size from seeding the output of Alg, consider a reduction from the following problem, which we call SEED-STAR:

PROBLEM 1 (SEED-STAR). The input graph is restricted to be a collection of k stars of size n/k each (as in Figure 3), and the algorithm observes a fixed number of influence sample from the input graph. Furthermore, if an influence sample contains the center of a star subgraph, then the entire star subgraph is revealed to the algorithm. Given the influence samples and revealed star subgraphs, the SEED-STAR problem is to choose k seeds such that their expected spread size is maximal.

Let Alg' be any optimal algorithm using less than βk^2 influence samples for the SEED-STAR problem. Consider the outputs of Alg and Alg' on the hard example in Figure 3. Because both Alg and Alg' seed optimally given βk^2 influence samples, and Alg' has strictly more information than Alg, the expected spread size from seeding the output of Alg' is at least as large as Alg; hence, it suffices to upper-bound the expected spread size from seeding the output of Alg'. Note that an optimal seed set of size k for Alg' should include the centers of all the revealed stars. If less than k stars are revealed, then it is optimal for Alg' to choose the remaining seeds randomly from among the sampled leaf nodes that do not belong to any of the revealed stars. Any such leaf node will contribute

$$1 + \frac{1}{k} + \frac{1}{k^2}\left(\frac{n}{k} - 2\right) < 1 + \frac{1}{k} + \frac{n}{k^3}.$$

Because there at most k such leaf nodes the total contribution of the leaf nodes to the expected spread size of the out of Alg' is at most $k + 1 + n/k^2$. Next we show that the probability of a new star being revealed to Alg' as a result of an influence sample is always strictly less than $2/k$. To see why, note that if the initial node of the influence sample belongs to a previously seen star then this probability is zero. Conditioned on the initial node belonging to a new star, then influence sample will reveal the star if it is its center node, which happens with probability k/n, or if it is a leaf node and its incident edge is activated, which happens with probability at most $1/k$. Hence, the probability of a new start being revealed at any influence sample is upper-bounded by

$$\frac{k}{n} + \frac{1}{k} < \frac{2}{k},$$

where the last inequality is because of the assumption $n > k^2$. Therefore, the expected number of stars that are revealed by βk^2 influence samples is at most $2\beta k$. Any such star contributes n/k^2 to expected spread size of the output of Alg'. We can thus upper-bound the expected spread size from seeding the output of Alg', as therefore Alg, as follows:

$$(2\beta k)\frac{n}{k^2} + k + 1 + \frac{n}{k^2} < ((2\beta + 1)k + 2)\frac{n}{k^2} = \mu\mathsf{OPT} - \epsilon n,$$

using $n > k^2$ in the last inequality. The proof follows as with Alg being optimal, no algorithms can provide an approximation guarantee that exceeds $\mu\mathsf{OPT} - \epsilon n$, using less than βk^2 influence samples on this example.

B.4. Proof of Theorem 4: Hardness of approximation with edge queries

We present our hard example for $k = 1$ 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 $(v_1, u_1), (v_2, u_2), ...(v_{3/\mu}, u_{3/\mu})$ be the list of the selected edges. we remove $(v_1, u_1), (v_2, u_2), ...(v_{3/\mu}, u_{3/\mu})$ and replace them by (u_1, v_2), (u_2, v_3), ..., $(u_{3/\mu-1}, v_{3/\mu})$, $(u_{3/\mu}, v_1)$. Note that this process connects all of the selected clusters while preserving the node degree (see Figure 5).

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 $\mathsf{OPT} = \mu n/3$ nodes. Next we show that the expected spread size form seeding the output of Alg on G' is less than $\mu^2 n/3$, which means that Alg is not an 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 result 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

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

Moreover, note that the run of Alg on G and G' is the same unless Alg queries one of the positions (i.e., edges) that we change in G to construct G'. In what follows, we upper-bound the probability that Alg queries one of the changed positions in G by $\mu/3$. This implies that Alg cannot be a μ-approximation because its expected spread size is strictly less than $\mu\mathsf{OPT}$:

$$\left(1 - \frac{\mu}{3}\right)\frac{2\mu^2 n}{9} + \frac{\mu}{3}\frac{3}{\mu}\frac{\mu^2 n}{9} < \frac{\mu^2 n}{3} = \mu\mathsf{OPT}.$$

To finish the proof, we need to bound the probability that Alg queries one of the changed positions in G. Let A_i be the (possibly random) number of edges that 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, with probability one (with respect to randomness of Alg), we have $A_i < (\mu^3/27)\binom{\mu^2 n/9}{2}$. Therefore, the total probability (with respect to randomness of Alg and G') that Alg queries a changed position in the i-th clique is upper-bounded by

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

By a union bound over all $9/\mu^2$ cliques we can upper-bound the probability that Alg queries a changed position by $(9/\mu^2)(\mu^3/27) = \mu/3$, as claimed. This completes the proof of the theorem.

B.5. Proof of Lemma 1: Combining additive and multiplicative approximation loss

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

$$\Gamma(\Lambda'_\delta) + \epsilon n \geq \Gamma_\delta(\Lambda'_\delta), \tag{8}$$

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

$$\Gamma_\delta(\Lambda'_\delta) \geq \alpha \Gamma_\delta(\Lambda) - \beta n \geq \alpha \Gamma(\Lambda) - (\beta + \alpha\epsilon)n \tag{9}$$

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 (8) and (9) to get that $\Gamma(\Lambda'_\delta) \geq \alpha \Gamma(\Lambda) - (\beta + (\alpha+1)\epsilon)n$.

B.6. Proof of Lemma 2: Sub-sampling the node set

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

$$|\Gamma_\rho(S) - \Gamma(S)| \leq \epsilon n. \tag{10}$$

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

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

Hoeffding-Bernstein inequality (van der Vaart and Wellner 1996, Lemma 2.14.19) provides that

$$\mathbb{P}\left[\left|\frac{1}{n\rho}\sum_{v \in \mathcal{V}_\rho}\phi(v,S) - \frac{1}{n}\Gamma(S)\right| \geq \epsilon\right] = \mathbb{P}\left[|\Gamma_\rho(S) - \Gamma(S)| \geq \epsilon n\right]$$

$$\leq 2\exp\left(-\frac{2n\rho\epsilon^2}{2\sigma_n^2 + \epsilon\Delta_n}\right), \tag{11}$$

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

$$\sigma_n^2 = \frac{1}{n}\sum_{v \in \mathcal{V}}\left(\phi(v,S) - \frac{1}{n}\Gamma(S)\right)^2$$

$$= \frac{1}{n}\sum_{v \in \mathcal{V}}\phi(v,S)^2 - \left(\frac{1}{n}\Gamma(S)\right)^2$$

$$\leq \frac{1}{n}\sum_{v \in \mathcal{V}}\phi(v,S) - \left(\frac{1}{n}\Gamma(S)\right)^2$$

$$= \ell - \ell^2 \leq \ell \leq 1.$$

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 (11) subsequently simplifies

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

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

$$\mathbb{P}\left[|\Gamma_\rho(\mathcal{S}) - \Gamma(\mathcal{S})| \geq \epsilon n\right] \leq 2\exp(-(\delta' + \log 2)) = e^{-\delta'}. \tag{12}$$

To complete the proof we use a union bound to claim that (10) 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 (12).

B.7. Proof of Lemma 3: Probing the extended neighborhoods

We begin by considering a fixed $\mathcal{S} \subset \mathcal{V}$. Recall $\Gamma_\rho^{(T)}(\mathcal{S}) = (1/\rho)\sum_{v \in \mathcal{V}_\rho} \phi^{(T)}(v, \mathcal{S})$ and $\phi^{(T)}(v, \mathcal{S}) = (1/T)\sum_{i=1}^{T} Y^{(i)}(v, \mathcal{S})$ for $v \in \mathcal{V}_\rho$. When $v \in \mathcal{V}_\rho$ is fixed, the Bernoulli variables $Y^{(1)}(v, \mathcal{S}), \ldots, Y^{(T)}(v, \mathcal{S})$ are independent and identically distributed with mean $\phi(v, \mathcal{S})$. By Chernoff bound to $\phi^{(T)}(v, \mathcal{S})$, we get that:

$$\mathbb{P}\left[\left|\phi^{(T)}(v, \mathcal{S}) - \phi(v, \mathcal{S})\right| > \epsilon\right] \leq 2\exp(-\epsilon^2 T/3), \text{ for any fixed } v \in \mathcal{V}_\rho.$$

Using $T = T_{\epsilon,\delta}^{n,k}$, by union bound over the choice of $\binom{n}{k}$ seed sets $\mathcal{S} \subset \mathcal{V}$, $\mathrm{card}(\mathcal{S}) = k$, 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 $v \in \mathcal{V}_\rho$:

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

B.8. Proof of Lemma 4: Limiting the size of the probed neighborhoods

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

It is useful to think of $\mathcal{G}_{\rho,\tau}^{(1)}, \ldots, \mathcal{G}_{\rho,\tau}^{(T)}$ as subgraphs of $\mathcal{G}_{\rho}^{(1)}, \ldots, \mathcal{G}_{\rho}^{(T)}$. An immediate consequence of this observation is that for any set of nodes $\mathcal{S}$, we have $\Gamma_{\rho}^{(T)}(\mathcal{S}) \geq \Gamma_{\rho,\tau}^{(T)}(\mathcal{S})$. We call the imaginary process whereby $\mathcal{G}_{\rho,\tau}^{(i)}$ is obtained after removing some nodes and edges from $\mathcal{G}_{\rho}^{(i)}$ a τ-cutting, and subsequently, we refer to $\mathcal{G}_{\rho,\tau}^{(i)}$ and $\mathcal{G}_{\rho}^{(i)}$ as the cut and uncut copies, respectively. Finally, it is also useful to define $\phi_{\tau}^{(T)}(v,\mathcal{S})$ in the exact same way as (3) but using the τ-cut copies $\mathcal{G}_{\rho,\tau}^{(1)}, \ldots, \mathcal{G}_{\rho,\tau}^{(T)}$.

The proof follows (Bateni et al. 2017, Lemma 2.4) closely. In particular, we first note that it suffices to show the existence of a set $\mathcal{L}$, $\mathrm{card}(\mathcal{L}) = k$ satisfying $\Gamma_{\rho,\tau}^{(T)}(\mathcal{L}) \geq (1-\epsilon)\mathsf{OPT}_{\rho}^{(T)}$. Because if there exists such a set $\mathcal{L}$, then for any α-approximate solution $\Lambda_{\rho,\tau}^{\alpha,(T)}$ we can write (recall $\Gamma_{\rho}^{(T)}(\mathcal{S}) \geq \Gamma_{\rho,\tau}^{(T)}(\mathcal{S})$ for all $\mathcal{S}$):

$$\Gamma_{\rho}^{(T)}\left(\Lambda_{\rho,\tau}^{\alpha,(T)}\right) \geq \Gamma_{\rho,\tau}^{(T)}\left(\Lambda_{\rho,\tau}^{\alpha,(T)}\right) \geq \alpha\Gamma_{\rho,\tau}^{(T)}\left(\Lambda_{\rho,\tau}^{(T)}\right) \geq \alpha\Gamma_{\rho,\tau}^{(T)}\left(\mathcal{L}\right) \geq (1-\epsilon)\alpha\mathsf{OPT}_{\rho}^{(T)},$$

implying that $\Lambda_{\rho,\tau}^{\alpha,(T)}$ 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 $\mathbb{E}\left\{\Gamma_{\rho,\tau}^{(T)}(\mathbf{L})\right\} \geq (1-\epsilon)\mathsf{OPT}_{\rho}^{(T)}$. The set $\mathbf{L}$ is constructed is as follows: Starting from $\Lambda_{\rho}^{(T)}$, remove ϵk of its nodes randomly, and replace them with ϵk nodes chosen uniformly at random from $\mathcal{V}$. To see why $\mathbb{E}\left\{\Gamma_{\rho,\tau}^{(T)}(\mathbf{L})\right\} \geq (1-\epsilon)\mathsf{OPT}_{\rho}^{(T)}$, consider

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

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

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

It only remains to verify the truth of the former inequality, $\mathbb{E}\left\{\phi_{\tau}^{(T)}(v,\mathbf{L})\right\} \geq (1-\epsilon)\phi^{(T)}(v,\Lambda_{\rho}^{(T)}) + \epsilon$. First note that $\mathbb{E}\left\{\phi_{\tau}^{(T)}(v,\mathbf{L})\right\}$ represents the probability of node v being connected to one of the nodes in the random set $\mathbf{L}$ averaged over the T subsampled graphs $(\mathcal{G}_{\rho,\tau}^{(1)}, \ldots, \mathcal{G}_{\rho,\tau}^{(T)})$. Consider each of the T copies in our uncut sketch, $\mathcal{G}_{\rho}^{(1)}, \ldots, \mathcal{G}_{\rho}^{(T)}$, and the connections between node v and the optimal set $\Lambda_{\rho}^{(T)}$ in these uncut copies. If these connections remain unchanged in the τ-cut copies $\mathcal{G}_{\rho,\tau}^{(1)}, \ldots, \mathcal{G}_{\rho,\tau}^{(T)}$, 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 τ-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 $\tau_{\epsilon}^{n,k} = \lceil n\log(1/\epsilon)/(\epsilon k)\rceil$ nodes is chosen is upper-bounded by ϵ:

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

B.9. Proof of Lemma 5: Total approximation loss from subsampling and limited probing

Following the notation in the proof of Lemma 4 (Appendix B.8), consider any $\Lambda_{\rho,\tau}^{\alpha,(T)}$. Lemma 4 implies that $\Lambda_{\rho,\tau}^{\alpha,(T)}$ is also $\Lambda_{\rho}^{(1-\epsilon)\alpha,(T)}$ because the loss in approximation factor from limited probing is at most $(1-\epsilon)$ when $\tau = \tau_{\epsilon}^{n,k}$. Next note that Lemma 3, together with Lemma 1, 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: $\Gamma_{\rho}(\Lambda_{\rho}^{(1-\epsilon)\alpha,(T)}) \geq (1-\epsilon)\alpha\mathsf{OPT}_{\rho} - ((1-\epsilon)\alpha + 1)\epsilon n$. Finally, another application of Lemma 1 with Lemma 2 yields that with at least $1 - e^{-\delta}$ probability, $\Gamma(\Lambda_{\rho}^{(1-\epsilon)\alpha,(T)}) \geq (1-\epsilon)\alpha\mathsf{OPT} - 2((1-\epsilon)\alpha + 1)\epsilon n$. The proof is complete upon combining the preceding statements to get that, with total probability at least $1 - 2e^{-\delta}$, $\Gamma(\Lambda_{\rho,\tau}^{\alpha,(T)}) \geq (1-\epsilon)\alpha\mathsf{OPT} - 2((1-\epsilon)\alpha + 1)\epsilon n$.

B.10. Proof of Theorem 5: Upper-bounding the total number of edges queries

To begin, we bound the total number of edges used in our sketch, i.e., the T subsampled graphs $(\mathcal{G}_{\rho,\tau}^{(1)}, \ldots, \mathcal{G}_{\rho,\tau}^{(T)})$. 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 subsampled graphs. 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 $\mathrm{PROBE}(\rho, T, \tau, p)$. Using the Chernoff upper tail bound and the fact that $\mathbb{E}[\mathbf{X}] \leq p\binom{\tau}{2}$, we can upper-bound $\mathbf{X}$ as follows:

$$\mathbb{P}\left[\mathbf{X} \geq p\tau(\tau-1)/2 + \delta'\sqrt{p\tau(\tau-1)/2}\right] \leq \mathbb{P}\left[\mathbf{X} \geq \mathbb{E}[\mathbf{X}] + \delta'\sqrt{\mathbb{E}[\mathbf{X}]}\right]$$
$$\leq e^{-\delta'^2/4}. \tag{13}$$

Recall from (6) that $E_{\epsilon,p}^{n,k} = p\tau_{\epsilon}^{n,k}(\tau_{\epsilon}^{n,k} - 1)/2 = p\binom{\tau}{2}$ with $\tau = \tau_{\epsilon}^{n,k}$. Setting $\tau = \tau_{\epsilon}^{n,k}$ and

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

in (13) 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 subsampled graphs, we have $\mathbf{X} \leq \overline{X}$, where

$$\overline{X} := E_{\epsilon,p}^{n,k} + \sqrt{\delta(\tau_{\epsilon}^{n,k}\log n + \log T)E_{\epsilon,p}^{n,k}}. \tag{14}$$

Next note that starting from any of the $n\rho$ nodes in $\mathcal{V}_{\rho}$ we never hit more than τ nodes following the limited probing procedure — see the "while loop" condition in step 8 of the $\mathrm{PROBE}(\rho, T, \tau, p)$ algorithm. Moreover, the only way for the connected component of one of the $n\rho$ initial nodes in $\mathcal{G}_{\rho,\tau}^{(i)}$ to get larger than τ is if an edge is added which combines two connected components each containing at least one node in $\mathcal{V}_{\rho}$. Upon inclusion of any such edge in $\mathcal{G}_{\rho,\tau}^{(i)}$, no further edges will be added to the corresponding connected component because of the "while loop" condition in step 8 of PROBE. Hence, there could be at most $n\rho$ such edges in each of the subsampled graphs, and in total there are at most $n\rho T$ such edges in our sketch. Upon removing all such edges, any connected

component in the remainder of our sketch will have size at most τ and their total number is always less than $n\rho T$. Given (14), we can bound the total number of edges in our sketch by $n\rho T + n\rho T\overline{X}$. More precisely, with probability at least $1 - e^{-\delta}$, we have:

$$\text{card}(\mathcal{E}_T) \leq n\rho T(1 + \overline{X}) = C_{\epsilon,\delta}^{n,k}, \tag{15}$$

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 (6).

Note that during the construction of $\mathcal{G}_{\rho,\tau}^{(i)}$, if the revealed neighbor (ν_ι) in step 10 of the PROBE algorithm is previously probed, then the following "if statement" in step 11 prevents the queried edge from being added to $\mathcal{G}_{\rho,\tau}^{(i)}$. Such edges are queried but not added to $\mathcal{G}_{\rho,\tau}^{(i)}$ because they have already got their chance of appearing in $\mathcal{G}_{\rho,\tau}^{(i)}$ once (during the probing of ν_ι). 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 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 queried 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 $\mathcal{G}_{\rho,\tau}^{(i)}$. In (15) 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 subsampled graphs. 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:

$$\mathbb{P}\left[\sum_e A_e^{(i)} C_e^{(i)} \geq p\sum_e A_e^{(i)} + 2n\sqrt{\delta + \log T}\right]$$
$$\leq \exp\left(-\frac{4n^2(\delta + \log T)}{2\left(p\sum_e A_e^{(i)} + n\sqrt{\delta + \log T}\right)}\right)$$
$$\leq \exp\left(-\delta - \log T\right) = \frac{1}{T}e^{-\delta}, \tag{16}$$

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

$$\sum_e B_e^{(i)} = \sum_e A_e^{(i)} C_e^{(i)} \leq p\sum_e A_e^{(i)} + 2n\sqrt{\delta + \log T}. \tag{17}$$

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 $\mathcal{G}_{\rho,\tau}^{(i)}$, 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 (17) yields:

$$\sum_e B_e^{(i)} \leq p \sum_e D_e^{(i)} + 2n\sqrt{\delta + \log T}, \tag{18}$$

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 $\mathcal{G}_{\rho,\tau}^{(i)}$. Hence, from (15) we have:

$$\sum_i \sum_e E_e^{(i)} = \mathrm{card}(\mathcal{E}_T) \leq C_{\epsilon,\delta}^{n,k}, \tag{19}$$

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 (16), 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 (18) 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:

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

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

$$p \sum_e D_e^{(i)} \leq \sum_e E_e^{(i)} + n\sqrt{2(\delta + \log T)}. \tag{20}$$

Combing (18) and (20) and taking the summation over $i = 1, \ldots, T$ gives that with probability at least $1 - 2e^{-\delta}$:

$$\sum_i \sum_e B_e^{(i)} \leq \sum_i \sum_e E_e^{(i)} + \left(2 + \sqrt{2}\right) Tn\sqrt{\delta + \log T}. \tag{21}$$

To complete the proof, we combine (19) and (21) to get the claimed upper bound on the total number of edge queries:

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

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

B.11. Proof of Theorem 6: Lower-bounding the required number of edge queries

For this proof, we expand on our construction in Appendix B.4. We set $k = 1$ and prove the hardness result for fixed $0 < \mu < 1$ and $0 \le \epsilon < \mu^2/18$, while assuming that $3/\mu$, $1/\mu^2$, and $n\mu^2/9$ are all integers for simplicity, without any loss in generality. We present our hard example for

$$p > \frac{9}{\mu^2} \frac{\log n + c}{n} \text{ and } n > (18e^{\bar{c}}/\mu^2)^4, \tag{22}$$

where $\bar{c} = c + \log(\gamma\mu/3)/c$, c is a constant satisfying $c > \max\{5 + 2\log 2, 1 - \log(\gamma/6), -\log(\gamma\mu/3)\}$, and $0 < \gamma < \mu/6$ is a constant that is fixed arbitrarily. Note that p is allowed to vary with n subject to (22). As in Appendix B.4, we rely on a modification of a collection of $9/\mu^2$ cliques by connecting $3/\mu$ of them at random around a circle. One key difference in our construction here is that rather than connecting each clique to the next one by rewiring a single link (as in Figure 5), we do so using a collection of $-p^{-1}\log(\gamma\mu/6)$ edges, where $0 < \gamma < \mu/6$ is a fixed constant. Let graph G be the collection of $9/\mu^2$ isolated cliques, each of size $\mu^2 n/9$. We construct G', our hard input graph, from G via the following random process:

1. Select $3/\mu$ cliques at random and label them by $i = 1, 2, 3, \ldots, 3/\mu$.
2. Select one edge from each of the $3/\mu$ selected cliques uniformly at random. Let $(v_1, u_1), (v_2, u_2), \ldots (v_{3/\mu}, u_{3/\mu})$ be the list of the first $3/\mu$ selected edges.
3. Remove $(v_1, u_1), (v_2, u_2), \ldots (v_{3/\mu}, u_{3/\mu})$ and replace them by (u_1, v_2), (u_2, v_3), $\ldots$, $(u_{3/\mu-1}, v_{3/\mu})$, $(u_{3/\mu}, v_1)$. Note that this process connects all of the selected clusters while preserving the degree distribution (see Figure 5).
4. Repeat this process until $-p^{-1}\log(\gamma\mu/6)$ edges are chosen from each clique, i, and rewired to connect to the proceeding clique, $i + 1$: $1 \to 2 \to 3 \to \ldots \to i \to i + 1 \to \ldots \to 3/\mu \to 1$.

We refer to these $3/\mu$ cliques in graph G' as *rewired cliques*; see Figure 5. Let Alg be an arbitrary (potentially randomized) algorithm for influence maximization that queries less than $pC^\epsilon_{\mu,\gamma} n^2$ edges, where

$$C^\epsilon_{\mu,\gamma} = \frac{-(\mu^6 - 9\mu^4\epsilon)}{14580 \log(\gamma\mu/6)}.$$

We are interested in the run of Alg on G'. Note that with $k = 1$ the optimum on G' is to seed one of the $n\mu/3$ nodes in the rewired cliques. The gist of our proof here is the same as the one in Appendix B.4: we show that the optimum seed on G' induces an expected spread size of at least $(1 - \gamma)n\mu/3$; on the other hand, seeding the output of any algorithm that queries less than $pC^\epsilon_{\mu,\gamma} n^2$ edges of G', induces an expected spread size that is at most $(1 - \gamma)\mu^2 n/3 < \mu\text{OPT}$.

To begin, note that the active edges on each clique constitute an Erdős-Rényi random graph with edge probability p on $n\mu^2/9$ nodes, except for the $3/\mu$ rewired cliques. The edge probability, p,

satisfying (22) is large enough to induce a connected random graph on each of the $9/\mu^2 - 3/\mu$ isolated cliques (with high probability as $n \to \infty$). Similarly, since only, $-p^{-1}\log(\gamma\mu/6) \in O(n/\log n)$, of the $\binom{n\mu^2/9}{2}$ edges in each of the $3/\mu$ cliques are randomly chosen and rewired, the induced random graphs on each of the $3/\mu$ rewired cliques are also going to be connected with high probability as $n \to \infty$. Hence, any time that a node in any of the cliques is activated, all of the nodes in that clique are going to be activated. Formally, the probability that a pair of nodes in any of the clique are connected by an active edge can be lower-bounded as follows:

$$\frac{9(\log n + c)}{n\mu^2} - \frac{-p^{-1}\log(\gamma\mu/6)}{\binom{n\mu^2/9}{2}} = \frac{9(\log n + c)}{n\mu^2} + \frac{2(9/\mu^2)\log(\gamma\mu/3)}{(\log n + c)(n\mu^2/9 - 1)} \qquad n > e^c$$

$$> \frac{9(\log n + c)}{n\mu^2} + \frac{(9/c\mu^2)\log(\gamma\mu/3)}{(n\mu^2/9 - 1)} \qquad n > n\mu^2/9 - 1$$

$$> \frac{9(\log n + c)}{n\mu^2} + \frac{(9/c)\log(\gamma\mu/3)}{n\mu^2} = \frac{\log n + \bar{c}}{\bar{n}} =: \bar{p},$$

where $\bar{c} = c + \log(\gamma\mu/3)/c$ is a constant term that is corrected for the effect of the $-p^{-1}\log(\gamma\mu/6)$ edges that are removed at random from each of the $3/\mu$ rewired cliques and $\bar{n} = n\mu^2/9$ is the size of each clique. Note that by assumption we have $c > -\log(\gamma\mu/3)$ which implies that $c - 1 < \bar{c} < c$. These edges are used to construct the graph G' by connecting the rewired cliques together (Figure 5). Our next lemma allows us to upper-bound the probability that the active edges on a fixed clique do not constitute a connected graph.

LEMMA 6. *Let C be a random graph on $\bar{n}$ nodes with edge probability $\bar{p} = (\log n + \bar{c})/\bar{n}$, where $n = 9\bar{n}/\mu^2$, $0 < \mu < 1$, and $\bar{c} > 2(2 + \log 2)$ are fixed constants. Let $\bar{\mathcal{C}}$ be the event that C is not connected. If $n > (18e^{\bar{c}}/\mu^2)^4$, then $\mathbb{P}(\bar{\mathcal{C}}) < \mu^2 e^{-\bar{c}}/3$.*

Proof. We use a common technique in random graph theory to upper-bound the probability that a random graph on $\bar{n}$ nodes with edge probability $\bar{p}$ is not connected. Following (Bollobás 2001, Theorem 7.3), $\mathbb{P}(\bar{\mathcal{C}})$ is upper-bounded by the sum of the expected values of the number of the connected components of sizes $j = 1, 2, \ldots, \bar{n}/2$:

$$\mathbb{P}(\bar{\mathcal{C}}) < \sum_{j=1}^{\bar{n}/2} \mathbb{E}(X_j), \tag{23}$$

where X_j is a random variable that counts the number of connected components of size j in a random graph on $\bar{n}$ nodes with edge probability $\bar{p}$. We next bound the different terms in (23) individually. Starting with $j = 1$, we have:

$$\mathbb{E}(X_1) = \bar{n}(1 - \bar{p})^{\bar{n}-1} < \bar{n}\exp(-\bar{p}\bar{n}) = \frac{\mu^2 e^{-\bar{c}}}{9}. \tag{24}$$

For $j = 2$, we have:

$$\mathbb{E}(X_2) = \binom{\bar{n}}{2} \bar{p}(1-\bar{p})^{2(\bar{n}-2)} < \bar{p}\bar{n}^2 \exp(-2\bar{p}\bar{n})$$

$$= \frac{e^{-2\bar{c}}\mu^2}{9} \frac{\log n + \bar{c}}{n} \qquad\qquad \bar{c} < \log n$$

$$< \frac{e^{-2\bar{c}}\mu^2}{9} \frac{2\log n}{n} \qquad\qquad 2\log n/n \le 2/e < 1$$

$$< \frac{e^{-2\bar{c}}\mu^2}{9}. \tag{25}$$

For $3 \le j \le \bar{n}/2$, we can bound $\mathbb{E}(X_j)$ by the expected number of trees on j nodes as follows:

$$\mathbb{E}(X_j) \le \binom{\bar{n}}{j} j^{j-2} \bar{p}^{j-1}(1-\bar{p})^{j(\bar{n}-j)} < (\bar{n}e/j)^j j^{j-2} \bar{p}^{j-1}(1-\bar{p})^{j(\bar{n}-j)}$$

$$< j^{-2} \exp(j + j\log\bar{n} + (j-1)\log\bar{p} - \bar{p}j(\bar{n}-j)) \qquad\qquad j^{-2} < 1$$

$$< \exp(j(1-\bar{c}) - j(1-\mu^2/9)\log n + (j-1)\log\bar{p} + j^2\bar{p}) \qquad\qquad j^2\bar{p} < j(\log n + \bar{c})/2 \text{ for all } j \le \bar{n}/2$$

$$< \exp(j(1-\bar{c}/2) - (j/2 - \mu^2/9)\log n + (j-1)\log((\log n + \bar{c}))) \qquad\qquad \mu^2/9 < 1 \text{ and } \bar{c} < \log n$$

$$< \exp(j(1-\bar{c}/2) - (j-1)\log n/2 + (j-1)\log(2\log n))) \qquad\qquad -\log 2 < 0$$

$$< e^{j(1+\log 2 - \bar{c}/2)} \left(\log n/\sqrt{n}\right)^{j-1} \qquad\qquad 2(2 + \log 2) < c - 1 < \bar{c} \text{ and } 3 \le j$$

$$< e^{-j}(\log n)^2/n. \tag{26}$$

Replacing (24), (25), and (26) in (23) and using the fact that $\sum_{j=1}^{\bar{n}/2} e^{-j} < \frac{e}{e-1} < 2$ we get:

$$\mathbb{P}(\bar{\mathcal{C}}) < 2\frac{\mu^2 e^{-\bar{c}}}{9} + 2\frac{(\log n)^2}{n}$$

$$= 2\frac{\mu^2 e^{-\bar{c}}}{9} + \frac{(\log n)^2}{n^{3/4}} \frac{2}{n^{1/4}} \qquad\qquad \frac{(\log n)^2}{n^{3/4}} < 1$$

$$< 2\frac{\mu^2 e^{-\bar{c}}}{9} + \frac{2}{n^{1/4}} \qquad\qquad (18e^{\bar{c}}/\mu^2)^4 < n$$

$$< \frac{\mu^2 e^{-\bar{c}}}{3},$$

finishing the proof.

Fix any of the $9/\mu^2$ cliques and let $\bar{P}$ be the probability that it is not connected. Note that this probability is decreasing in the probability of the edges being active and because we lower-bounded the latter by $\bar{p}$, it suffices to upper-bound $\bar{P}$ assuming that the probability of having an active edge between each pair of nodes is $\bar{p}$. Equipped with Lemma 6, we know that $\bar{P} < \mu^2 e^{-\bar{c}}/3$ if $n > (18e^{\bar{c}}/\mu^2)^4$. Next, consider the event that the active edges induce a disconnected graph on at least one of the $9/\mu^2$ cliques and denote this event by $\mathcal{E}_1$. By a union bound over the $9/\mu^2$ cliques,

we obtain that $\mathbb{P}(\mathcal{E}_1) \le (9/\mu^2)\bar{P} < 3e^{-\bar{c}}$, which upon choosing $\bar{c} > -\log(\gamma/6)$ can be upper-bounded by $\gamma/2$:

$$\mathbb{P}(\mathcal{E}_1) < \frac{\gamma}{2}, \text{ for } n > (18e^{\bar{c}}/\mu^2)^4 \text{ and } \bar{c} > -\log(\gamma/6)\cdot \tag{27}$$

We next consider the event that for at least one connected pair of rewired cliques, none of the $-p^{-1}\log(\gamma\mu/6)$ rewired edges between them are active. Let us denote this event by $\mathcal{E}_2$. we upper-bound $\mathbb{P}(\mathcal{E}_2)$ by $\gamma/2$. Consider the $-p^{-1}\log(\gamma\mu/3)$ edges connecting the i-th rewired clique to the $(i+1)$-th rewired clique and let A_i be the random variable that is the number of active edges among these $-p^{-1}\log(\gamma\mu/3)$ edges. Random variable A_i has a Bernoulli distribution with parameters $-p^{-1}\log(\gamma\mu/3)$ and p. Subsequently, we can bound the probability that $A_i = 0$ as follows:

$$\mathbb{P}(A_i = 0) = (1-p)^{-p^{-1}\log(\gamma\mu/6)} < e^{\log(\gamma\mu/6)} = \frac{\gamma\mu}{6}.$$

By a union bound over the $3/\mu$ rewired cliques, we obtain that, with probability at most $\gamma/2$, there is a pair of rewired cliques that are connected together but none of the $-p^{-1}\log(\gamma\mu/6)$ edges that connect them are active:

$$\mathbb{P}(\mathcal{E}_2) < \gamma/2\cdot \tag{28}$$

Having thus upper-bounded both $\mathbb{P}(\mathcal{E}_1)$ and $\mathbb{P}(\mathcal{E}_2)$ by $\gamma/2$ in (27) and (28), we conclude that $\mathbb{P}(\mathcal{E}_1 \cup \mathcal{E}_2) < \mathbb{P}(\mathcal{E}_1) + \mathbb{P}(\mathcal{E}_2) < \gamma$. Subsequently, we can give a $1-\gamma$ lower bound on the probability of the complement event, $\mathcal{E}_1^c \cap \mathcal{E}_2^c$, that the active edges induce a connected graph on each of the cliques and at least one of the $-p^{-1}\log(\gamma\mu/6)$ rewired edges between each pair of connected cliques is active. Hence, with probability at least $1-\gamma$, after seeding one of the $n\mu/3$ nodes on the rewired cliques, the activation is going to spread from one rewired clique to the next. In other words, seeding one of rewired clique nodes activates all of the $n\mu/3$ nodes on the rewired cliques with probability at least $1-\gamma$, and the optimal expected spread size on G' is at least $(1-\gamma)n\mu/3$.

So far we have shown that the optimum on G' is at least $(1-\gamma)n\mu/3$. Next we show that the expected spread size from seeding the output of Alg on G' is less than $(1-\gamma)\mu^2 n/3$, which means that Alg cannot be a μ-approximation algorithm. This implies that there is no μ-approximation algorithm that queries less than $pC^\epsilon_{\mu,\gamma}n^2 \in O_\mu(pn^2)$ 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, Alg seeds one of the nodes of G at random. Observe that the expected spread size of a random seed in G' is upper-bounded as follows:

$$\left(1 - \frac{3/\mu}{9/\mu^2}\right)\frac{n\mu^2}{9} + \left(\frac{3/\mu}{9/\mu^2}\right)\frac{n\mu}{3} = \left(2 - \frac{\mu}{3}\right)\frac{\mu^2 n}{9} < 2(1-\gamma)\frac{\mu^2 n}{9}, \tag{29}$$

where the last inequality follows since $\gamma < \mu/6$. Moreover, note that the runs of Alg on G and G' are the same unless Alg queries one of the positions (i.e., edges) that we rewire in G to construct G'. Next we upper-bound the probability that Alg queries one of the rewired edges by $\mu/3 - 3\epsilon/\mu$. Using the upper bound in (29), this implies that the expected spread size from seeding the output of Alg on G' is at most $(1-\gamma)\mu^2 n/3 - \epsilon n$, as claimed:

$$
\left(1 - \frac{\mu}{3} + \frac{3\epsilon}{\mu}\right)\frac{2(1-\gamma)\mu^2 n}{9} + \left(\frac{\mu}{3} - \frac{3\epsilon}{\mu}\right)\frac{n\mu}{3} = \left(3 - 2\gamma - \frac{2\mu}{3}(1-\gamma) + \frac{6\epsilon}{\mu}(1-\gamma)\right)\frac{\mu^2 n}{9} - \epsilon n
$$
$$
< (3 - 2\gamma - \frac{\mu}{3}(1-\gamma))\frac{\mu^2 n}{9} - \epsilon n
$$
$$
< \frac{(1-\gamma)\mu^2 n}{3} - \epsilon n,
$$

where the penultimate is because $6\epsilon/\mu < \mu/3$ and the last inequality is because $\mu(1-\gamma)/3 > \gamma$. To see why, recall the assumptions $\epsilon < \mu^2/18$ and $\gamma < \mu/6$. In particular, with the latter we have $\gamma < \mu(1 - 1/6)/3 < \mu(1 - \mu/6)/3 < \mu(1-\gamma)/3$.

To finish the proof, it only remains to verify that $\mu/3 - 3\epsilon/\mu$ is an upper bound on the probability of Alg querying one of the rewired edges of G'. To this end, let $\mathcal{A}_i$ be the (possibly random) set of all edges belonging to the i-th clique in G that are queried by Alg. Conditioned on $\mathcal{A}_i$, consider any of the edges belong to $\mathcal{A}_i$. The probability that this specific edge is rewired when constructing G' from G is at most:

$$
\frac{-p^{-1}\log(\gamma\mu/6)}{\binom{\mu^2 n/9}{2}}.
$$

Note that this probability is conditioned on the realization of $\mathcal{A}_i$ and is with respect to the random process by which G' is constructed from G. Therefore, conditioned on $\mathcal{A}_i$, the probability that a rewired edge belonging to the i-th clique is queried by Alg is at most:

$$
\mathrm{card}(\mathcal{A}_i)\frac{-p^{-1}\log(\gamma\mu/6)}{\binom{\mu^2 n/9}{2}}\frac{3/\mu}{9/\mu^2} \leq \left(\frac{-pn^2(\mu^6 - 9\mu^4\epsilon)}{14580\log(\gamma\mu/6)}\right)\left(\frac{-\log(\gamma\mu/6)}{p\binom{\mu^2 n/9}{2}}\right)\frac{\mu}{3}
$$
$$
< \frac{n(\mu^2 - 9\epsilon)}{90(n-9)}\frac{\mu}{3} < \frac{\mu^3}{27} - \epsilon\frac{\mu}{3}.
$$

where, in the fist inequality, we have used the assumption that Alg queries less than $pC_{\mu,\gamma}^\epsilon n^2$ edges; therefore, $\mathrm{card}(\mathcal{A}_i) \leq pC_{\mu,\gamma}^\epsilon n^2$ with probability one. In the last inequality, we use the fact that $n/(90(n-9)) < 1/9$ for $n > 10$, and in particular, for $n > (18e^{\bar{c}}/\mu^2)^4 > 18^4$ in (22). By averaging over all realization of $\mathcal{A}_i$ and a union bound over all of the $9/\mu^2$ cliques, we can bound the probability that Alg queries any rewired edge by $(9/\mu^2)(\mu^3/27 - \epsilon\mu/3) = \mu/3 - 3\epsilon/\mu$, completing the proof.

B.12. Proof of Theorem 7: Approximation guarantees with bounded edge queries

Given $\epsilon' > 0$, fix $\epsilon = \epsilon'/7$, $\delta = 2\log n$ and set $\rho = \rho_{\epsilon,\delta}^{n,k}$, $T = T_{\epsilon,\delta}^{n,k}$ and $\tau = \tau_{\epsilon}^{n,k}$ according to Algorithm 4. First we consider the approximation guarantee of $\text{SEED}(\epsilon, k)$. Running Algorithm $\text{PROBE}(\rho, T, \tau, p)$, provides access to the submodular function $\Gamma_{\rho,\epsilon}^{(T)}$ which has the k-IM optimal solution $\Lambda_{\rho,\epsilon}^{(T)}$ with value $\text{OPT}_{\rho,\epsilon}^{(T)}$. Using $\text{SEED}(\epsilon, k)$, we obtain an approximate solution to k-IM on $\Gamma_{\rho,\epsilon}^{(T)}$. Call this solution $\Lambda_{\rho,\epsilon}^{\star(T)}$. The analysis of (Mirzasoleiman et al. 2015, Theorem 1) implies that

$$\mathbb{E}\left[\Gamma_{\rho,\epsilon}^{(T)}(\Lambda_{\rho,\epsilon}^{\star(T)})\right] \geq (1 - 1/e - \epsilon)\Gamma_{\rho,\epsilon}^{(T)}(\Lambda_{\rho,\epsilon}^{(T)})$$
$$= (1 - 1/e - \epsilon)\text{OPT}_{\rho,\epsilon}^{(T)},$$

where the expectation is with respect to the randomness of the SEED algorithm. We can combine the claims of Lemma 5 and Theorem 5 with the assumption $n \geq (30/\epsilon')^2 > \sqrt{35/\epsilon'} = \sqrt{5/\epsilon}$ to guarantee that with probability at least $1 - 5e^{-\delta} = 1 - 5/n^2 \geq 1 - \epsilon$ we have

$$\Gamma(\Lambda_{\rho,\epsilon}^{(T)}) = \text{OPT}_{\rho,\epsilon}^{(T)} \geq (1 - \epsilon)\text{OPT} - 2(2 - \epsilon)\epsilon n.$$

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:

$$\mathbb{E}\left[\Gamma(\Lambda_{\rho,\epsilon}^{\star(T)})\right] \geq (1 - 1/e - \epsilon)\mathbb{E}\left[\text{OPT}_{\rho,\epsilon}^{(T)}\right]$$
$$\geq (1 - 1/e - \epsilon)(1 - \epsilon)((1 - \epsilon)\text{OPT} - 2(2 - \epsilon)\epsilon n)$$
$$\geq (1 - 1/e - \epsilon)(1 - 2\epsilon)\text{OPT} - 4\epsilon n$$
$$\geq (1 - 1/e)(1 - 2\epsilon)\text{OPT} - (\epsilon)(1 - 2\epsilon)\text{OPT} - 4\epsilon n$$
$$\geq (1 - 1/e)\text{OPT} - 7\epsilon n = (1 - 1/e)\text{OPT} - \epsilon' n,$$

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

Next we analyze the running times of $\text{PROBE}(\rho, T, \tau, p)$ and $\text{SEED}(\epsilon, k)$ algorithms. Let us denote the expected number of edge queries by q. Using Theorem 5, we can upper bound q as follows:

$$q \leq (1 - 3e^{-\delta})Q_{\epsilon,\delta}^{n,k} + 3e^{-\delta}(2Tn^2) \leq Q_{\epsilon,\delta}^{n,k} + 6T$$
$$\in O_{\epsilon'}(pn^2\log^4 n + \sqrt{kp}n^{1.5}\log^{5.5} n + kn\log^{3.5} n), \tag{30}$$

where $Q_{\epsilon,\delta}^{n,k}$ is given by (7) with $\epsilon = \epsilon'/7$ and $\delta = 2\log n$; and the $2Tn^2$ upper bound is by probing every node (so that each edge is queried twice). Note that the $n \geq \sqrt{\delta + \log T}$ condition in Theorem 5 is always satisfied for $n \geq (30/\epsilon')^2$ with our choice of δ and T. Because with $\delta = 2\log n$ we have $\log T = \log(3(\delta + \log 2)(k + 1)\log n/\epsilon^2) < \log(18n\log^2 n/\epsilon^2) < 4\log n$ for all $n \geq (30/\epsilon')^2 > 18/\epsilon^2$; therefore, $\sqrt{\delta + \log T} < \sqrt{6\log n} < n$, the later being true for all $n \geq 3$.

Every iteration of the "while loop" in step 8 of the PROBE algorithm leads to an edge query, and therefore, the expected run time of $\text{PROBE}(\rho, T, \tau, p)$ is $O(q)$. Note that the expected number of edges in the T subsampled graphs is upper bounded by q. Hence, using any typical graph traversal algorithm (such as DFS or BFS), we can identify the connected components of all the T subsampled graphs in a time that is $O(q)$. To compute $\Delta(v|\Lambda^\star)$, we go over the connected components of ν 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 its number of initial nodes (belonging to $\mathcal{V}_\rho$). After adding a node v to $\Lambda^\star$, we reset the value of the connected components containing v in each of the T subsampled graphs to zero (see steps 10 and 11 of the SEED algorithm). This ensures that if we later pick another node from these components we do not double count those initial nodes that are already covered by v. This process can be done in $O(T) = O_{\epsilon'}(k \log^2 n)$ time, where $T = T_{\epsilon,\delta}^{n,k}$ in (5) with $\epsilon = \epsilon'/7$ and $\delta = 2 \log n$. Finally, note that as step 9 of Algorithm 3 is repeated k times, the submodular maximization algorithm that we are using makes no more than $k \cdot \text{card}(\mathcal{R}) = n \log(1/\epsilon) = n \log(7/\epsilon') = O_{\epsilon'}(n)$ queries to the function $\Delta(\cdot|\cdot)$. Combining the preceding bounds with (30) puts the total running time of SEED and PROBE algorithms in $O_{\epsilon'}(pn^2 \log^4 n + \sqrt{kp}n^{1.5} \log^{5.5} n + kn \log^{3.5} n)$ as claimed.

B.13. Proof of Theorem 8: Hardness of approximation with query discrepancy

Consider the probe and seed cascade probabilities, and denote their minimum and maximum by $\check{p} := \min\{p, p'\}$ and $\hat{p} := \max\{p, p'\}$; by assumption, we have $\hat{p}/\check{p} = 1 + \delta$. We show when the query and seed cascade probability are different (given by p and p'), then for ϵ satisfying

$$\epsilon < \frac{8\mu(\delta - \delta^2)c_\delta}{41(2 - \mu) + 16(\delta - \delta^2)c_\delta} < \frac{\mu}{2}, \quad \text{where } c_\delta = 1 + (\delta - \delta^2)/2, \tag{31}$$

there can be no k-IM approximation algorithm that guarantees an expected spread size of at least $\mu\mathsf{OPT} - \epsilon n$ on any input graph with optimum expected spread size OPT. Let Alg be any such k-IM approximation algorithm. Define $n_{\epsilon,\mu} := (1 - 2\epsilon/\mu)n$ and note that under (31) we have $2\epsilon/\mu < 1$. The input graph for our hard example is comprised of two subgraphs: $\mathcal{G}_1$ and $\mathcal{G}_2$. The former is a clique of size $2\epsilon n/\mu$ and the latter is a random graph with edge probability $(1 - \delta/2)/(n_{\epsilon,\mu}\check{p})$ on the remaining $n_{\epsilon,\mu}$ nodes. We prove our hardness results for $k = 1$ under the following specifications:

$$n > \frac{256}{(\epsilon - 3\mu e^{-c} - \delta^2)^2}, \quad \text{and} \tag{32}$$

$$\check{p} = \min\{p, p'\} > \frac{\log n + c}{2\epsilon n/\mu}, \quad \text{where} \tag{33}$$

$$c = \max\left\{2(2 + \log 2), \log\left(\frac{3\mu}{3 - \delta^2}\right)\right\}.$$

First note that the expected spread size when seeding one of the $\mathcal{G}_1$ clique nodes is at most $2\epsilon n/\mu$, irrespective of whether the cascade probability is p or p'. We use Lemma 6, with $\bar{n} = 2\epsilon n/\mu$ and $\bar{p} = \check{p}$

satisfying (33), to provide a companion lower bound on the expected spread size when seeding $\mathcal{G}_1$. Accordingly, if $n > (\mu e^c/\epsilon)^4$, then the probability that active edges on $\mathcal{G}_1$ do not form a connected graph is at most $3\mu e^{-c}/(2\epsilon)$. Having established this lower bound for $\check{p} = \min\{p, p'\}$, it holds true for either of the two cascade probabilities (p and p') because increasing the cascade probability can only increase the expected spread size.

So far we have shown that the expected spread size from seeding one of the $\mathcal{G}_1$ nodes is at most $2\epsilon n/\mu$ and at least

$$\left(1 - \frac{3\mu e^{-c}}{2\epsilon}\right)\frac{2\epsilon n}{\mu} = \frac{2\epsilon n}{\mu} - 3ne^{-c},$$

both bounds being true irrespective of whether the cascade probability is p or p'. We next show that an opposite situation holds in case of $\mathcal{G}_2$. The expected spread size from seeding one of the nodes in $\mathcal{G}_2$ depends critically on the cascade probability and can be much larger or much smaller than the range that we have established for $\mathcal{G}_1$. We use techniques from the analysis of the giant connected component in random graphs (Janson et al. 2011, Theorem 5.4) to lower-bound the expected spread size from seeding $\mathcal{G}_2$ when the cascade probability is $\hat{p}$ and to upper-bound it when the cascade probability is $\check{p}$. Comparing the two bounds with the range that we have established for the expected spread size on $\mathcal{G}_1$ reveals that Alg should necessarily seed $\mathcal{G}_2$ when cascade probability is $\hat{p}$ and $\mathcal{G}_1$ when the cascade probability is $\check{p}$. Therefore, there can be no approximation algorithm that provides a $\mu\mathsf{OPT} - \epsilon n$ guarantee while the query and seed cascade probabilities are different. We show this for the small enough ϵ in (31), based on the discrepancy between the query and seed cascade probabilities ($\hat{p}/\check{p} = 1 + \delta$).

We begin by lower-bounding the expected spread size from seeding a random node in $\mathcal{G}_2$ when the cascade probability is $\hat{p}$. Note that if the cascade probability is $\hat{p}$, then the active edges after the independent cascade on $\mathcal{G}_2$ constitute a random graph with edge probability:

$$\hat{p}\frac{1 - \delta/2}{n_{\epsilon,\mu}\check{p}} = \frac{(1 + \delta)(1 - \delta/2)}{n_{\epsilon,\mu}} = \frac{c_\delta}{n_{\epsilon,\mu}} > \frac{1}{n_{\epsilon,\mu}}.$$

In this case we show that the expected spread size from seeding a random node in $\mathcal{G}_2$ is at least $n_{\epsilon,\delta}/2$, where

$$n_{\epsilon,\delta} := \frac{16n_{\epsilon,\mu}(\delta - \delta^2)c_\delta}{41}.$$

We do so by upper-bounding the probability that a random node on $\mathcal{G}_2$ belongs to a component of size less than $n_{\epsilon,\delta}$. Let us denote this event by $\mathcal{E}$. To upper-bound $\mathbb{P}(\mathcal{E})$, we consider a random node ν and search the component containing ν according to the following procedure:

PROCEDURE 1. Starting from ν, initialize the set of discovered nodes to include only ν and set $X_0 = 1$. At step one ($i = 1$), declare all neighbors of ν to be discovered and set X_1 equal to the number of newly discovered nodes. Also declare ν to be explored. At any step $i > 1$, choose an unexplored node at random from the set of discovered but unexplored nodes; declare it to be explored; add all of its undiscovered neighbors to the discovered set; and set X_i equal to the number of newly discovered nodes at step i.

FACT 1. Consider $\tau \geq 1$ iterations of Procedure 1. The total number of discovered but unexplored nodes after τ iterations is given by: $\sum_{i=0}^{\tau} X_i - \tau$. Therefore, at any step τ, the size of the connected component containing ν is greater than τ if, and only if, $\sum_{i=0}^{\tau} X_i > \tau$; the size of the connected component containing ν is equal to τ if, and only if, $\sum_{i=0}^{\tau} X_i = \tau$; and the size of the connected component containing ν is less than τ if, and only if, $\sum_{i=0}^{\tau} X_i < \tau$.

Fact 1 implies that $\mathbb{P}(\mathcal{E}) = \mathbb{P}\left(\sum_{i=0}^{n_{\epsilon,\delta}} X_i < n_{\epsilon,\delta}\right)$. To upper-bound $\mathbb{P}(\mathcal{E})$, let X_i^- be i.i.d. random variables with their common distribution set to:

$$\text{Binomial}\left(n_{\epsilon,\mu} - n_{\epsilon,\delta}, \frac{c_\delta}{n_{\epsilon,\mu}}\right).$$

Note that for all $i \leq n_{\epsilon,\delta}$, X_i stochastically dominates X_i^-; hence,

$$\mathbb{P}(\mathcal{E}) = \mathbb{P}\left(\sum_{i=0}^{n_{\epsilon,\delta}} X_i < n_{\epsilon,\delta}\right) \leq \mathbb{P}\left(\sum_{i=0}^{n_{\epsilon,\delta}} X_i^- \leq n_{\epsilon,\delta}\right). \tag{34}$$

To ease the notation, define $X^- = \sum_{i=0}^{n_{\epsilon,\delta}} X_i^-$ and note that X^- also has a binomial distribution with parameters given by:

$$\text{Binomial}\left(n_{\epsilon,\delta}(n_{\epsilon,\mu} - n_{\epsilon,\delta}), \frac{c_\delta}{n_{\epsilon,\mu}}\right).$$

We next point out that,

$$\begin{aligned}
\mathbb{E}[X^-] &= n_{\epsilon,\delta}(n_{\epsilon,\mu} - n_{\epsilon,\delta})\frac{c_\delta}{n_{\epsilon,\mu}} \\
&= n_{\epsilon,\delta}\left(1 - \frac{16}{41}(\delta - \delta^2)c_\delta\right)c_\delta \\
&= n_{\epsilon,\delta}\left(1 + \frac{\delta - \delta^2}{2}(1 - \frac{32}{41}c_\delta^2)\right) \\
&> n_{\epsilon,\delta},
\end{aligned} \tag{35}$$

where the last inequality follows because $0 < \delta - \delta^2 \leq 1/4$ which implies $c_\delta = 1 + (\delta - \delta^2)/2 \leq 9/8$ and $32c_\delta^2/41 \leq 81/82 < 1$, for all $0 < \delta < 1$. Combining (34) and (35), we get:

$$\mathbb{P}(\mathcal{E}) \leq \mathbb{P}\left(X^- \leq n_{\epsilon,\delta}\right) \leq \mathbb{P}\left(X^- < \mathbb{E}[X^-]\right) < \frac{1}{2}.$$

We have established that the probability of the event $\mathcal{E}$ that the size of the connected component of a random node in $\mathcal{G}_2$ is less than $n_{\epsilon,\delta}$ is at most $1/2$. This implies that the expected spread size from seeding a random node in $\mathcal{G}_2$ is at least $n_{\epsilon,\delta}/2$. Under (31), we have $2\epsilon n/\mu < \mu n_{\epsilon,\delta}/2 - \epsilon n$; hence, whenever the cascade probability is $\hat{p}$, Alg should necessarily seed one of the nodes in $\mathcal{G}_2$.

Next, we upper-bound the expected spread size from seeding a node in $\mathcal{G}_2$ when the cascade probability is $\check{p}$. For a fixed node ν in $\mathcal{G}_2$, let $\bar{\mathcal{E}}$ be the event that the size of the connected component containing ν is greater than

$$\bar{n}_{\epsilon,\delta} := \frac{16 \log n_{\epsilon,\mu}}{\delta^2}.$$

Let X_i be defined in the same way as in Procedure 1, then Fact 1 continues to hold and we have:

$$\mathbb{P}(\bar{\mathcal{E}}) = \mathbb{P}\left(\sum_{i=0}^{\bar{n}_{\epsilon,\delta}} X_i > \bar{n}_{\epsilon,\delta}\right).$$

To upper-bound $\mathbb{P}(\bar{\mathcal{E}})$, let X_i^+ be i.i.d. random variables with common Binomial$(n_{\epsilon,\mu}, (1-\delta/2)/n_{\epsilon,\mu})$ distribution and note that for all i, X_i^+ stochastically dominates X_i; hence,

$$\begin{aligned}
\mathbb{P}(\bar{\mathcal{E}}) &= \mathbb{P}\left(\sum_{i=0}^{\bar{n}_{\epsilon,\delta}} X_i > \bar{n}_{\epsilon,\delta}\right) \\
&\leq \mathbb{P}\left(\sum_{i=0}^{\bar{n}_{\epsilon,\delta}} X_i^+ > \bar{n}_{\epsilon,\delta}\right) \\
&= \mathbb{P}\left(\sum_{i=1}^{\bar{n}_{\epsilon,\delta}} X_i^+ \geq \bar{n}_{\epsilon,\delta}(1-\delta/2) + \bar{n}_{\epsilon,\delta}\delta/2\right) \\
&\leq \exp\left(\frac{-\bar{n}_{\epsilon,\delta}^2 \delta^2/4}{2\bar{n}_{\epsilon,\delta}(1-\delta/2) + \bar{n}_{\epsilon,\delta}\delta/3}\right) \\
&\leq \exp\left(\frac{-\bar{n}_{\epsilon,\delta}\delta^2}{8}\right) = \frac{1}{n_{\epsilon,\mu}^2},
\end{aligned}$$

where the penultimate inequality follows by applying a Chernoff bound to $\sum_{i=0}^{\bar{n}_{\epsilon,\delta}} X_i^+$ which is a binomial random variable with mean $\bar{n}_{\epsilon,\delta}(1-\delta/2)$. By union bound, the probability of having at least one node that belongs to a connected component of size greater than $\bar{n}_{\epsilon,\delta}$ is at most $n_{\epsilon,\mu}\mathbb{P}(\bar{\mathcal{E}}) = 1/n_{\epsilon,\mu}$; or equivalently, with probability at least $1 - 1/n_{\epsilon,\mu}$, all of the nodes in $\mathcal{G}_2$ belong to connected components of size less than or equal to $\bar{n}_{\epsilon,\delta}$. Hence, when the cascade probability is $\check{p}$, the expected spread size from seeding any of the nodes in $\mathcal{G}_2$ is at most

$$\left(1 - \frac{1}{n_{\epsilon,\mu}}\right)\bar{n}_{\epsilon,\delta} + \frac{1}{n_{\epsilon,\mu}}n_{\epsilon,\mu} < \bar{n}_{\epsilon,\delta} + 1.$$

Therefore, whenever cascade probability is $\check{p}$ and

$$\bar{n}_{\epsilon,\delta} + 1 < \mu(2\epsilon n/\mu)(1 - 3\mu e^{-c}/(2\epsilon)) - \epsilon n = (\epsilon - 3\mu e^{-c})n,$$

Alg would necessarily seed one of the nodes in $\mathcal{G}_1$ and achieve $(2\epsilon n/\mu)(1 - 3\mu e^{-c}/(2\epsilon))$ expected spread size. The conditions in (32) are sufficient to ensure $\bar{n}_{\epsilon,\delta} + 1 < (\epsilon - 3\mu e^{-c})n$. This completes

the proof showing that any approximation algorithm Alg seeds different nodes for different cascades probabilities on our hard example. Therefore, the approximation guarantees cannot be achieved when the query and seed probability differ under the specified conditions.

B.14. A pruning algorithm to correct for query discrepancy

Algorithm 5: $\mathrm{PRUNE}(p, p')$

Input: $\mathcal{G}'^{(1)}_{\rho,\tau}, \ldots, \mathcal{G}'^{(T)}_{\rho,\tau}$ and $\mathcal{V}_\rho$ generated by $\mathrm{PROBE}(\rho, T, \tau, p')$, and target cascade probability p

Output: $\mathcal{G}^{(1)}_{\rho,\tau}, \ldots, \mathcal{G}^{(T)}_{\rho,\tau}$ simulating the output of $\mathrm{PROBE}(\rho, T, \tau, p)$

1 Require: $p' > p$.

2 Set $h = p/p'$.

3 for i **from** 1 **to** T **do**

4   Initialize $\mathcal{G}^{(i)}_{\rho,\tau} \leftarrow \mathcal{G}'^{(i)}_{\rho,\tau}$.

5   **for** *every edge e in $\mathcal{G}^{(i)}_{\rho,\tau}$* **do**

6   Let H be an independent Bernoulli draw with $\mathbb{P}[H = 1] = h$.

7   **if** $H = 0$ **then**

8   Remove e from $\mathcal{G}^{(i)}_{\rho,\tau}$.

9   **end**

10   **end**

11   **for** *every node ν in $\mathcal{G}^{(i)}_{\rho,\tau}$* **do**

12   **if** *none of the nodes in $\mathcal{V}_\rho$ are reachable from ν* **then**

13   Remove ν from $\mathcal{G}^{(i)}_{\rho,\tau}$.

14   **end**

15   **end**

16 end

17 return $\mathcal{G}^{(1)}_{\rho,\tau}, \ldots, \mathcal{G}^{(T)}_{\rho,\tau}$.

Appendix C: Extensions to other influence models

We presented our results for undirected graphs with a homogeneous cascade probability (p). In subsection C.1, we present the extension of our results to directed influence graphs. In subsection C.2, we explore other influence models beyond the independent cascade and provide a pathway to perform edge queries in the general class of triggering influence models.

C.1. Directed influence graphs

To perform edge queries on directed graphs, in steps 6 and 7 of the PROBE algorithm we let $\mathcal{N}_\nu$ denote the set of *incoming* neighbors (i.e., nodes that influence ν) such that the edge query in step 10 reveals the ι-th incoming neighbor of ν. Two directed edges that are between the same pair of nodes in opposite directions ($u \to \nu$ and $\nu \to u$) 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 $\mathcal{G}_{\rho,\tau}^{(i)}$. Subsequently, the "if statement" in step 11 of the PROBE algorithm should be removed when performing edge queries on directed graphs: any queried edge in step 10 is always added to $\mathcal{G}_{\rho,\tau}^{(i)}$ in the following steps. As a further consequence, our edge query upper bound in the directed case consists entirely of the edges that appear in the sketch (denoted by $\mathcal{E}_T$ in Appendix B.10) and is given by (15): with probability at least $1 - e^\delta$ no more than $C_{\epsilon,\delta}^{n,k} \in O_{\epsilon,\delta}(pn^2 \log^2 n + \sqrt{kp}n^{1.5}\log^{2.5} n + k^2 \log^2 n)$ edges are queried — a slight improvement over our edge query upper bound in the undirected case.

In the "while loop" condition in step 8 of the PROBE algorithm, instead of considering the size of the connected component of node ν, we count the number of nodes that are reachable via directed paths from ν (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.
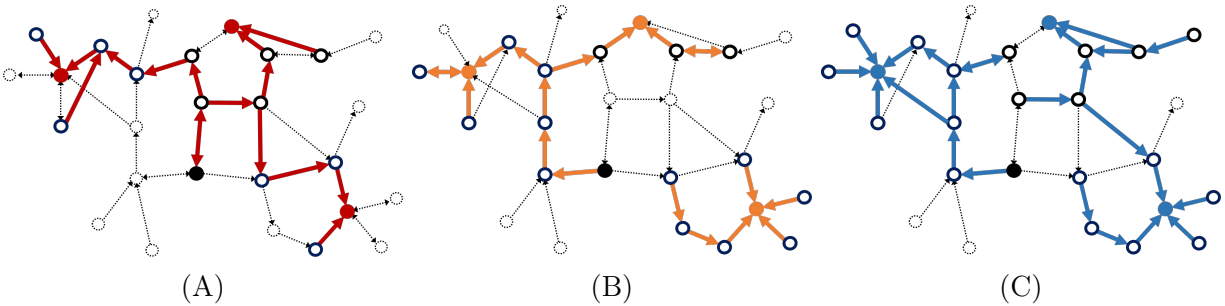

(A) (B) (C)

Figure 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 unsampled nodes and edges are dotted.

We need to make similar changes to the way candidates are scored in the SEED algorithm. First, in step 3 of SEED we set the value of each initial node (belonging to $\mathcal{V}_\rho$) to be one. When evaluating

the marginal increments $(\Delta(v|\Lambda^\star))$ in step 7 of SEED, we add the values of the initial nodes that are reachable via directed paths from v in each of the T subsampled graphs $(\mathcal{G}_{\rho,\tau}^{(1)}, \ldots, \mathcal{G}_{\rho,\tau}^{(T)})$, rather than summing the value of its connected components (see Figure 8). Finally, in step 11 of SEED if an initial node is reachable from the chosen seed, then we nullify its value for scoring the subsequent candidates.
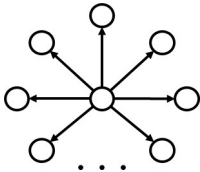

Figure 9   With $o(n)$ influence samples, one is unlikely to discover the center of the directed star network. In such a case it is impossible to guarantee a $\mu\mathsf{OPT} - \epsilon n$ expected spread size with fewer than $\Omega(n)$ queries.

In contrast, finding an approximately optimal seed set by influence sampling in a directed graph is very hard. For example, consider a star graph with n leafs where all of the edges are directed away from the center toward the leafs (see Figure 9). Assume that the cascade probability on each edge is 1. In this case, if we seed a leaf we only observe an isolated node and hence we need $\Omega(n)$ influence samples 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 reverse influence sampling as Borgs et al. (2014) do), 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 reverse influence samples 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). Following Algorithm 1 and Theorem 2, we can collect $O_\epsilon(k^2 \log n)$ reverse influence samples and choose the best k seeds by approximating the greedy steps.

C.2. Triggering models

In general, the influence graph may be directed and cascade probabilities may differ in each direction and across the edges. When performing edge queries, the probed nodes should reveal each of their incoming neighbors (influencers), with the cascade probability associated with that edge. Here we explain how a triggering set technique that is proposed by Kempe et al. (2015) helps us devise edge queries in a large class of influence models, including the independent cascade and *linear threshold* models. 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 et al. (2003, 2005, 2015) — through a conjecture that is positively resolved by Mossel and Roch (2010)

— 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.

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 — see, e.g., (Kempe et al. 2015, Section 3.2) and (Schoenebeck and Tao 2019). To avoid the intractable settings, Kempe et al. (2003, 2005, 2015) consider a randomized model where thresholds are i.i.d. uniform $[0, 1]$ variables. Mossel and Roch (2010) 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 (Mossel and Roch 2010, 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. (2003, 2015) 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 (Kempe et al. 2015, 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 cascade probability associated with the $u \to v$ edge (p_{uv}), independently at random. In the case of the linear threshold model, Kempe et al. (2015) 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 $\mathcal{T}_v = \varnothing$ 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. 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 C.1 — 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.
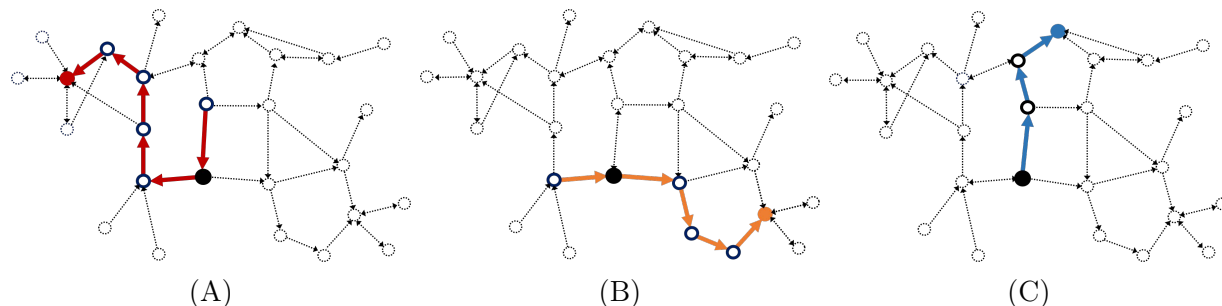
Figure 10   Some diffusion processes can be reduced to a triggering set model; whereby, after randomly drawing a triggering set from among her neighbors, a node becomes active if any of the nodes in her triggering set are active. In such models, we can devise edge queries by having a probed node reveal 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 the particular case of the linear threshold model, the proof of Theorem 2, and its adaptation to directed graphs in Appendix C.1, 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 any of the already chosen seeds. Following Theorem 2, we can provide a $(1 - 1/e)\mathsf{OPT} - \epsilon n$ approximation guarantee, by running $k\rho = k\lceil 81k \log(\frac{6nk}{\epsilon})/\epsilon^3 \rceil$ 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 $nk\rho = nk\lceil 81k \log(\frac{6nk}{\epsilon})/\epsilon^3 \rceil$. In Theorem 9, we state our results formally for the case of k-IM with the linear threshold model over directed graphs.

References

Akbarpour M, Malladi S, Saberi A (2020) Just a few seeds more: Value of network information for diffusion. Technical report, Stanford University, available at SSRN 3062830.

Alon N, Fischer E, Krivelevich M, Szegedy M (2000) Efficient testing of large graphs. *Combinatorica* 20(4):451–476.

Alon N, Fischer E, Newman I, Shapira A (2009) A combinatorial characterization of the testable graph properties: It's all about regularity. *SIAM Journal on Computing* 39(1):143–167.

Badanidiyuru A, Vondrák J (2014) Fast algorithms for maximizing submodular functions. *Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms*, 1497–1514 (Portland, Oregon, USA: SIAM).

Balkanski E, Immorlica N, Singer Y (2017a) The importance of communities for learning to influence. *Advances in Neural Information Processing Systems 30*, 5862–5871 (Long Beach, California, USA: Curran Associates, Inc.).

Balkanski E, Rubinstein A, Singer Y (2016) The power of optimization from samples. *Advances in Neural Information Processing Systems*, 4017–4025 (Barcelona, Spain: Curran Associates, Inc.).

Balkanski E, Rubinstein A, Singer Y (2017b) The limitations of optimization from samples. *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, 1016–1027 (Montreal, Canada: ACM).

Banerjee A, Chandrasekhar AG, Duflo E, Jackson MO (2013) The diffusion of microfinance. *Science* 341(6144):1236498.

Banerjee A, Chandrasekhar AG, Duflo E, Jackson MO (2019) Using gossips to spread information: Theory and evidence from two randomized controlled trials. *The Review of Economic Studies* 86(6):2453–2490.

Bateni M, Esfandiari H, Mirrokni V (2018) Optimal distributed submodular optimization via sketching. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1138–1147, KDD '18 (New York, NY, USA: ACM), ISBN 978-1-4503-5552-0.

Bateni MH, Esfandiari H, Mirrokni V (2017) Almost optimal streaming algorithms for coverage problems. *Proceedings of the 29th ACM Symposium on Parallelism in Algorithms and Architectures*, 13–23, SPAA '17 (New York, NY, USA: ACM), ISBN 978-1-4503-4593-4.

Behrman JR, Kohler HP, Watkins SC (2002) Social networks and changes in contraceptive use over time: Evidence from a longitudinal study in rural Kenya. *Demography* 39(4):713–738.

Bollobás B (2001) *Random Graphs* (Cambridge, UK: Cambridge University Press).

Borgs C, Brautbar M, Chayes J, Lucier B (2014) Maximizing social influence in nearly optimal time. *Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 946–957 (Portland, Oregon, USA: SIAM).

Cai J, De Janvry A, Sadoulet E (2015) Social networks and the decision to insure. *American Economic Journal: Applied Economics* 7(2):81–108.

Catanese SA, De Meo P, Ferrara E, Fiumara G, Provetti A (2011) Crawling Facebook for social network analysis purposes. *Proceedings of the International Conference on Web Intelligence, Mining and Semantics*, 1–8.

Chami GF, Ahnert SE, Kabatereine NB, Tukahebwa EM (2017) Social network fragmentation and community health. *Proceedings of the National Academy of Sciences* 114(36):E7425–E7431.

Chazelle B, Rubinfeld R, Trevisan L (2005) Approximating the minimum spanning tree weight in sublinear time. *SIAM Journal on computing* 34(6):1370–1379.

Chen W, Lin T, Tan Z, Zhao M, Zhou X (2016) Robust influence maximization. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 795–804, KDD '16 (San Francisco, California, USA: ACM).

Chen W, Wang Y, Yang S (2009) Efficient influence maximization in social networks. *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 199–208, KDD '09 (Paris, France: ACM).

Chin A, Eckles D, Ugander J (2021) Evaluating stochastic seeding strategies in networks. *Management Science* .

Christakis NA, Fowler JH (2010) Social network sensors for early detection of contagious outbreaks. *PloS One* 5(9):e12948.

Cohen E, Delling D, Pajor T, Werneck RF (2014) Sketch-based influence maximization and computation: Scaling up with guarantees. *Proceedings of the 23rd International Conference on Conference on Information and Knowledge Management*, 629–638 (Shanghai, China: ACM).

Cohen R, Havlin S, Ben-Avraham D (2003) Efficient immunization strategies for computer networks and populations. *Physical Review Letters* 91(24):247901.

Domingos P, Richardson M (2001) Mining the network value of customers. *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 57–66 (San Jose, California, USA: ACM).

Eckles D, Esfandiari H, Mossel E, Rahimian MA (2019) Seeding with costly network information. *Proceedings of the 2019 ACM Conference on Economics and Computation*, 421–422, EC '19 (New York, NY, USA: Association for Computing Machinery).

Esfandiari H, Mitzenmacher M (2018) Metric sublinear algorithms via linear sampling. *59th Annual Symposium on Foundations of Computer Science (FOCS)*, 11–22 (Paris, France: IEEE).

Feld SL (1991) Why your friends have more friends than you do. *American Journal of Sociology* 96(6):1464–1477.

Feng C, Fu L, Jiang B, Zhang H, Wang X, Tang F, Chen G (2020) Neighborhood matters: Influence maximization in social networks with limited access. *IEEE Transactions on Knowledge and Data Engineering* .

Flodgren G, Parmelli E, Doumit G, Gattellari M, O'Brien MA, Grimshaw J, Eccles MP (2011) Local opinion leaders: Effects on professional practice and health care outcomes. *The Cochrane Database of Systematic Reviews* 2011(8):CD000125.

Godes D, Mayzlin D (2009) Firm-created word-of-mouth communication: Evidence from a field test. *Marketing Science* 28(4):721–739.

Goel S, Salganik MJ (2010) Assessing respondent-driven sampling. *Proceedings of the National Academy of Sciences* 107(15):6743–6747.

Golovin D, Krause A (2011) Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research* 42:427–486.

Gomez-Rodriguez M, Leskovec J, Krause A (2012) Inferring networks of diffusion and influence. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 5(4):1–37.

Gonen M, Ron D, Shavitt Y (2011) Counting stars and other small subgraphs in sublinear-time. *SIAM Journal on Discrete Mathematics* 25(3):1365–1411.

Goyal A, Bonchi F, Lakshmanan LVS (2010) Learning influence probabilities in social networks. *Proceedings of the 3rd ACM International Conference on Web Search and Data Mining*, 241–250 (New York, New York, USA: ACM).

He X, Kempe D (2016) Robust influence maximization. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 885–894, KDD '16 (New York, NY, USA: ACM).

He X, Kempe D (2018) Stability and robustness in influence maximization. *ACM Transactions on Knowledge Discovery from Data* 12(6):66:1–66:34.

Heckathorn DD, Cameron CJ (2017) Network sampling: From snowball and multiplicity to respondent-driven sampling. *Annual Review of Sociology* 43:101–119.

Hinz O, Skiera B, Barrot C, Becker JU (2011) Seeding strategies for viral marketing: An empirical comparison. *Journal of Marketing* 75(6):55–71.

Horel T, Singer Y (2015) Scalable methods for adaptively seeding a social network. *Proceedings of the 24th International Conference on World Wide Web*, 441–451 (Florence, Italy: International World Wide Web Conference).

Indyk P (1999) Sublinear time algorithms for metric space problems. *Proceedings of the 31st Annual ACM Symposium on Theory of Computing*, 428–434 (Atlanta, Georgia, USA: ACM).

Janson S, Luczak T, Rucinski A (2011) *Random graphs*, volume 45 (Hoboken, NJ: John Wiley & Sons).

Karimi M, Lucic M, Hassani H, Krause A (2017) Stochastic submodular maximization: The case of coverage functions. *Advances in Neural Information Processing Systems*, 6853–6863 (Long Beach, CA, USA: Curran Associates, Inc.).

Kempe D, Kleinberg J, Tardos É (2003) Maximizing the spread of influence through a social network. *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 137–146 (Washington, DC, USA: ACM).

Kempe D, Kleinberg J, Tardos É (2005) Influential nodes in a diffusion model for social networks. *Proceedings of the 32nd International Colloquium on Automata, Languages, and Programming*, 1127–1138 (Lisbon, Portugal: Springer).

Kempe D, Kleinberg J, Tardos É (2015) Maximizing the spread of influence through a social network. *Theory of Computing* 11(4):105–147.

Kim DA, Hwong AR, Stafford D, Hughes DA, O'Malley AJ, Fowler JH, Christakis NA (2015) Social network targeting to maximise population behaviour change: A cluster randomised controlled trial. *The Lancet* 386(9989):145–153.

Kumar R, Moseley B, Vassilvitskii S, Vattani A (2015) Fast greedy algorithms in mapreduce and streaming. *ACM Transactions on Parallel Computing (TOPC)* 2(3):14.

Kumar V, Krackhardt D, Feld S (2018) Network interventions based on inversity: Leveraging the friendship paradox in unknown network structures, working Paper, Yale University.

Kumar V, Sudhir K (2019) Can friends seed more buzz and adoption?, Cowles Foundation Discussion Paper No. 2178.

Lattanzi S, Singer Y (2015) The power of random neighbors in social networks. *Proceedings of the 8th ACM International Conference on Web Search and Data Mining*, 77–86, WSDM '15 (Shanghai, China: ACM).

Leskovec J, Faloutsos C (2006) Sampling from large graphs. *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 631–636 (Philadelphia, PA, USA: ACM).

Libai B, Muller E, Peres R (2013) Decomposing the value of word-of-mouth seeding programs: Acceleration versus expansion. *Journal of Marketing Research* 50(2):161–176.

Lungeanu A, McKnight M, Negron R, Munar W, Christakis NA, Contractor NS (2021) Using Trellis software to enhance high-quality large-scale network data collection in the field. *Social Networks* 66:171–184.

Manshadi V, Misra S, Rodilitz S (2020) Diffusion in random networks: Impact of degree distribution. *Operations Research* 68(6):1722–1741.

Mihara S, Tsugawa S, Ohsaki H (2015) Influence maximization problem for unknown social networks. *Proceedings of the International Conference on Advances in Social Networks Analysis and Mining*, 1539–1546 (Paris, France: IEE/ACM).

Mihara S, Tsugawa S, Ohsaki H (2017) On the effectiveness of random jumps in an influence maximization algorithm for unknown graphs. *International Conference on Information Networking (ICOIN)*, 395–400 (Da Nang, Vietnam: IEEE).

Mirzasoleiman B, Badanidiyuru A, Karbasi A, Vondrák J, Krause A (2015) Lazier than lazy greedy. *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, 1812–1818 (Austin, Texas, USA: AAAI).

Mossel E, Roch S (2010) Submodularity of influence in social networks: From local to global. *SIAM Journal on Computing* 39(6):2176–2188.

Nemhauser GL, Wolsey LA, Fisher ML (1978) An analysis of approximations for maximizing submodular set functions–I. *Mathematical programming* 14(1):265–294.

Nguyen HT, Thai MT, Dinh TN (2016) Stop-and-stare: Optimal sampling algorithms for viral marketing in billion-scale networks. *Proceedings of the 2016 International Conference on Management of Data*, 695–710.

Paluck EL, Shepherd H, Aronow PM (2016) Changing climates of conflict: A social network experiment in 56 schools. *Proceedings of the National Academy of Sciences* 113(3):566–571.

Sadeh G, Cohen E, Kaplan H (2020) Sample Complexity Bounds for Influence Maximization. Vidick T, ed., *11th Innovations in Theoretical Computer Science Conference (ITCS 2020)*, volume 151 of *Leibniz International Proceedings in Informatics (LIPIcs)*, 29:1–29:36 (Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik).

Salganik MJ, Heckathorn DD (2004) Sampling and estimation in hidden populations using respondent-driven sampling. *Sociological Methodology* 34(1):193–240.

Schoenebeck G, Tao B (2019) Beyond worst-case (in)approximability of nonsubmodular influence maximization. *ACM Trans. Comput. Theory* 11(3):12:1–12:56, ISSN 1942-3454.

Seeman L, Singer Y (2013) Adaptive seeding in social networks. *Proceedings of the 54th Annual Symposium on Foundations of Computer Science (FOCS)*, 459–468 (Berkeley, California, USA: IEEE).

Stein S, Eshghi S, Maghsudi S, Tassiulas L, Bellamy RKE, Jennings NR (2017) Heuristic algorithms for influence maximization in partially observable social networks. *Proceedings of the 3rd International Workshop on Social Influence Analysis*, 20–32 (Melbourne, Australia: CEUR Workshop Proceedings).

Tang Y, Shi Y, Xiao X (2015) Influence maximization in near-linear time: A martingale approach. *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, 1539–1554.

Tang Y, Xiao X, Shi Y (2014) Influence maximization: Near-optimal time complexity meets practical efficiency. *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, 75–86.

Traud AL, Mucha PJ, Porter MA (2012) Social structure of Facebook networks. *Physica A: Statistical Mechanics and its Applications* 391(16):4165–4180.

van der Vaart AW, Wellner JA (1996) *Weak Convergence and Empirical Processes: With Applications to Statistics* (New York, NY, USA: Springer-Verlag).

Wang C, Chen W, Wang Y (2012) Scalable influence maximization for independent cascade model in large-scale social networks. *Data Mining and Knowledge Discovery* 25(3):545–576.

Wen Z, Kveton B, Valko M, Vaswani S (2017) Online influence maximization under independent cascade model with semi-bandit feedback. *Advances in Neural Information Processing Systems 30*, 3022–3032 (Long Beach, California, USA: Curran Associates, Inc.).

Wilder B, Immorlica N, Rice E, Tambe M (2017a) Influence maximization with an unknown network by exploiting community structure. *Proceedings of the 3rd International Workshop on Social Influence Analysis*, 2–7 (Melbourne, Australia: CEUR Workshop Proceedings).

Wilder B, Immorlica N, Rice E, Tambe M (2018) Maximizing influence in an unknown social network. *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, 4743–4750 (New Orleans, Louisiana, USA: AAAI).

Wilder B, Yadav A, Immorlica N, Rice E, Tambe M (2017b) Uncharted but not uninfluenced: Influence maximization with an uncertain network. *Proceedings of the 16th Conference on Autonomous Agents and Multiagent Systems*, 1305–1313 (São Paulo, Brazil: International Foundation for Autonomous Agents and Multiagent Systems).

Wu Q, Li Z, Wang H, Chen W, Wang H (2019) Factorization bandits for online influence maximization. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 636–646.

Yadav A, Wilder B, Rice E, Petering R, Craddock J, Yoshioka-Maxwell A, Hemler M, Onasch-Vera L, Tambe M, Woo D (2017) Influence maximization in the field: The arduous journey from emerging to deployed application. *Proceedings of the 16th Conference on Autonomous Agents and Multiagent Systems*, 150–158.

Dean Eckles   is an associate professor at the MIT Sloan School of Management. His research interests include social influence, networks, and causal inference.

Hossein Esfandiari   is a senior research scientist at Google. His works are in theoretical computer science, including approximation algorithms and sublinear-time algorithms.

Elchanan Mossel   works in probability, combinatorics, and inference. He is on the senior faculty of the Mathematics Department, with a jointly core faculty appointment at the Statistics and Data Science Center of Massachusetts Institute of Technology's Institute for Data, Systems, and Society.

M. Amin Rahimian   is an assistant professor of Industrial Engineering at University of Pittsburgh. He works at the intersection of networks, data, and decision sciences. He borrows tools from applied probability, statistics, algorithms, as well as decision and game theory to address problems of distributed inference and decentralized interventions in large-scale sociotechnical systems.