Influence analysis is a fundamental problem in social network analysis and mining. The important applications of the influence analysis in social network include influence maximization for viral marketing, finding the most influential nodes, online advertising, etc. For many of these applications, it is crucial to evaluate the influenceability of a node. In this paper, we study the problem of evaluating influenceability of nodes in social network based on the widely used influence spread model, namely, the independent cascade model. Since this problem is \#P-complete, most existing work is based on Naive Monte-Carlo (NMC) sampling. However, the NMC estimator typically results in a large variance, which significantly reduces its effectiveness. To overcome this problem, we propose two families of new estimators based on the idea of stratified sampling. We first present two basic stratified sampling (BSS) estimators, namely \textit{BSS-I} estimator and \textit{BSS-II} estimator, which partition the entire population into $2^r$ and $r+1$ strata by choosing $r$ edges respectively. Second, to further reduce the variance, we find that both \textit{BSS-I} and \textit{BSS-II} estimators can be recursively performed on each stratum, thus we propose two recursive stratified sampling (RSS) estimators, namely \textit{RSS-I} estimator and \textit{RSS-II} estimator. Theoretically, all of our estimators are shown to be unbiased and their variances are significantly smaller than the variance of the NMC estimator. Finally, our extensive experimental results on both synthetic and real datasets demonstrate the efficiency and accuracy of our new estimators. Additional Key Words and Phrases: Influenceability, influence networks, independent cascade model, Stratified sampling, uncertain graph

ACM Reference Format:
R-H. Li, J. Yu, Z. Shang. Estimating Node Influenceability in Social Networks. ACM V, N, Article A (January 2012), 26 pages.
DOI = 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

1. INTRODUCTION

Large scale online social networks (OSNs) such as Facebook and Twitter have become increasingly popular in the last years. Users in OSNs are able to share thoughts, activities, photos, and other information with their friends. As a result, the OSNs become an important medium for information dissemination and influence spread. A fundamental problem in such OSNs is to analyze and study the social influence among users [Tang et al. 2009]. Important applications of influence analysis in OSNs include influence maximization for viral marking [Kempe et al. 2003; Chen et al. 2010], finding the most influential nodes [Liu et al. 2009; Lappas et al. 2010], online advertising, etc. Especially, the influence maximization problem has recently attracted tremendous attention in research community [Leskovec et al. 2007; Chen et al. 2009; Chen et al. 2010; Goyal et al. 2011]. For many of these applications, a very important step is to accurately evaluate the influenceability of a node in OSNs.

The influenceability evaluation problem is based on influence spread in a network. Generally, the influence spread in a network can be modeled as a stochastic cascade model. In the literature, a widely used cascade mode is the independent cascade (IC) model. In the IC model, each node $i$ has a single chance to influence his/her neighbor $j$ with a probability $p_{ij}$, and such “influence event” is independent of the other “influence events” over other nodes. Due to the independent property, the IC model can be
represented by the probabilistic graph model, where each edge in the graph is associated with a probability and the existence of an edge is independent of any other edges [Potamias et al. 2010]. In this paper, we focus on the IC model and assume that the influence probabilities of all the edges in a social network are given in advance. In addition, we use the probabilistic graph model to represent the IC model.

This problem is equivalent to calculate the expected number of nodes in \( G \) that are reachable from \( s \), which is known to be \#P-complete [Chen et al. 2010]. The existing algorithms for this problem are based on naive Monte-Carlo sampling (NMC) [Kempe et al. 2003; Kempe et al. 2005; Chen et al. 2009]. However, NMC may result in a large variance, which significantly reduces its effectiveness. We will discuss this issue in detail in Section 3.

Given the IC model and a seed node \( s \), the influenceability evaluation problem is to compute the expected influence spread by the seed node \( s \). This problem is equivalent to calculate the expected number of nodes in a probabilistic graph \( G \) that are reachable from \( s \), which is known to be \#P-complete [Chen et al. 2010]. As a result, there is no hope to exactly evaluate the influenceability in polynomial time unless P=\#P. The existing algorithm for this problem is based on Naive Monte-Carlo sampling [Kempe et al. 2003; Kempe et al. 2005; Chen et al. 2009]. As our analysis given in Section 3 the Naive Monte-Carlo (NMC) estimator leads to a large variance, and thus it significantly reduces the effectiveness of the estimator. Theoretically, the NMC estimator can achieve arbitrarily close approximation to the exact value of the influenceability. However, this requires a large number of samples. Since performing a Monte-Carlo estimation needs to flip \( m \) coins to determine all the \( m \) edges of the network, the NMC estimator is extremely expensive to get a meaningful approximation of the influenceability in large networks. Consequently, the key issue to accelerate the NMC estimator is to reduce the number of samples that are needed to achieve a good accuracy.

In order to reduce the number of samples used in the NMC estimator, one potential solution is to reduce its variance. In this paper, we propose two types of the Monte-Carlo estimator, namely type-I estimator and type-II estimator, based on the idea of stratified sampling. All of our proposed estimators are shown to be unbiased and their variance are significantly smaller than the variance of the NMC estimator. To the best of our knowledge, this is the first work that addresses and studies the variance problem in NMC for influenceability evaluation problem.

To develop new type-I estimators, we devise an exact divide-and-conquer enumeration algorithm. Our exact algorithm starts by enumerating \( r \) edges, thus resulting in \( 2^r \) cases. Then, for each case the algorithm recursively enumerates another \( r \) edges. The recursion will terminate after all the \( m \) edges are enumerated. This exact algorithm has exponential time complexity to evaluate node's influenceability. Based on the exact algorithm, we propose a basic stratified sampling (BSS) estimator, namely BSS-I estimator, to estimate a node's influenceability. In particular, we first select \( r \) edges and determine their statuses (existence or inexistence). Obviously, this process generates \( 2^r \) cases. Then, we let each case be a stratum, and draw samples separately from each stratum. By carefully allocating the sample size for each stratum, we prove that the variance of the BSS-I estimator is smaller than the variance of the NMC estimator. Interestingly, we find that our BSS-I estimator can be recursively performed in each stratum, and thereby we propose a recursive stratified sampling estimator, namely RSS-I estimator. Since the RSS-I estimator recursively reduces the variance in each stratum, its variance is significantly smaller than the variance of the BSS-I

---

1 Learning the influence probabilities is out of scope of this paper. In the literature, there are some studies, such as [Goyal et al. 2010], on learning the influence probabilities in social network.
In addition to the type-I estimators (BSS-I and RSS-I), we further develop two type-II estimators based on a new stratification method. The new stratification method partitions the population into \( r + 1 \) strata by picking \( r \) edges. In the first stratum which is denoted by stratum 0, we set the statuses of all the \( r \) edges to "0", which denotes the edge inexistence. In the \( i \)-th (\( 1 \leq i \leq r \)) stratum, we set the statuses of all the first \( i - 1 \) edges to "0", the \( i \)-th edge to "1", which signifies the edge existence, and the rest \( r - i \) edges to "\(*\)", which denotes the status of the edge to be determined. Based on such stratification approach, we propose a basic stratified sampling estimator, namely BSS-II estimator. Similar to the idea of the RSS-I estimator, we develop a recursive stratified sampling estimator based on BSS-II estimator, namely RSS-II estimator. We conduct extensive experimental studies on both synthetic and real datasets, and we show that both RSS-I and RSS-II estimators reduce the variance of the NMC estimator significantly.

Note that the stratification approach in both type-I and type-II estimators are based on the \( r \) selected edges. Thus, an edge-selection strategy may significantly affect the performance of the estimators. In this paper, we present two edge-selection strategies for the proposed estimators: random edge-selection and Breadth-First-Search (BFS) edge-selection. The random edge-selection is to pick \( r \) unsampled edges randomly for stratification, while the BFS edge-selection picks \( r \) unsampled edges according to their BFS visiting order (the BFS starts from the seed node \( s \)). In our experiments, we show that an estimator with the BFS edge-selection strategy significantly outperforms the same estimator with the random edge-selection strategy.

Besides the influenceability estimation problem in social networks, our proposed estimation methods can be applied in many other application domains. For example, consider an application in a communication network with link failure. Given a router \( s \), it needs to count the expected number of hosts in the network that are reachable from \( s \). Such count assists network resource planning, and is also useful for network resource estimation, for example in P2P networks. Our proposed algorithms can provide accurate estimators for such application domains. In addition, our influenceability estimation methods can be directly used to the so-called influence function evaluation problem [Kempe et al. 2003], in which the seed is not only one node but a set of nodes. We can solve this problem by adding a virtual node \( s \) and link it to the set of seed nodes. Finally, our proposed stratified sampling estimators are very general, and can be easily used to handle uncertain graph mining problems, such as network reliability estimation [Rubino 1999], shortest path [Potamias et al. 2010], and reachability computation problem [Jin et al. 2011b].

The rest of this paper is organized as follows. We give the problem statement in Section 2 and introduce the Naive Monte-Carlo estimator in Section 3. We propose the type-I and type-II estimators in Section 4 and Section 5, respectively. Extensive experimental studies are reported in Section 6. Section 7 discusses the related work and Section 8 concludes this work.

2. PROBLEM STATEMENT
We consider a social network \( G = (V, E) \), where \( V \) denotes a set of nodes and \( E \) denotes a set of directed edges between the nodes. Let \( n = |V| \) and \( m = |E| \) be the number of nodes and edges in \( G \), respectively. In a social network, users (nodes) can perform actions, and the actions can propagate over the network. For example, in Twitter, an action denotes a user posts a tweet, and the action propagation denotes the event that the same tweet is re-posted (retweeted) by his/her followers. In this paper, we adopt the independent cascade (IC) model [Kempe et al. 2003] [Kempe et al. 2005] to model
such action propagation process. In the IC model, every edge \((u, v)\) is associated with an influence probability \(p_{uv}\) (Fig. 1(a), which represents the probability that a node \(v\) performs an action followed by the same action taken by its adjacent node \(u\). We refer to a social network \(G\) with influence probabilities as an influence network denoted by \(\mathcal{G} = (V, E, P)\), where the set \(P\) represents the set of influence probabilities. We call a node an active node if it performs an action.

The propagation process of the IC model unfolds in discrete steps. More precisely, we assume that a node \(v\) follows a node \(u\), and at step \(t\) node \(u\) performs an action \(\alpha\) and node \(v\) does not. Then, node \(u\) is given a single chance to influence node \(v\), and it succeeds with probability \(p_{uv}\). This probability is independent of other nodes that attempt to influence node \(v\). If node \(u\) succeeds, then node \(v\) will perform action \(\alpha\) at step \(t + 1\). In other words, node \(v\) is influenced by node \(u\) at step \(t + 1\). It is important to note that whether \(u\) succeeds or not, it cannot make any attempts to influence \(v\) again. The process terminates when there is no new node can be influenced.

The IC model can be initiated by a single node \(s\) such that the node performs an action before any other nodes in \(V \setminus \{s\}\). The seed node \(s\) models the source of influence, and it can spread across the network following the IC model. The propagation process is a stochastic process, after the process terminates, the number of active nodes is a random variable. Therefore, we take the expectation of this random variable to measure the influence spread of \(s\), and it is denoted as \(F_s(\mathcal{G})\). We refer to the expected influence spread of \(s\) (i.e. \(F_s(\mathcal{G})\)) as the influenceability of node \(s\). In this paper, we aim to evaluate the influenceability \(F_s(\mathcal{G})\) given a seed node \(s\). In the following subsection, we will give a formal definition of \(F_s(\mathcal{G})\) based on the probabilistic graph model [Potamias et al. 2010].

2.1. Influenceability Evaluation

Based on the IC model, an influence network \(\mathcal{G} = (V, E, P)\) is represented by the probabilistic graph model [Potamias et al. 2010], where the existence of an edge is independent of any other edges.

Given an influence network \(\mathcal{G} = (V, E, P)\), we denote a possible graph \(G_P = (V_P, E_P)\) which can be obtained by sampling each edge \(e\) in \(\mathcal{G}\) according to the influence probability \(p_e\) associated with the edge \(e\) \((p_e \in P)\). Here, we have \(V = V_P, E_P \subseteq E\), and the possible graph \(G_P\) has the probability \(\Pr[G_P]\), which is given by

\[
\Pr[G_P] = \prod_{e \in E_P} p_e \prod_{e \in E \setminus E_P} (1 - p_e).
\]

The total number of such possible graphs is \(2^m\), where \(m\) is the number of edges in \(\mathcal{G}\). For example, in Fig. 1(a) the influence network \(\mathcal{G}\) has \(2^{10}\) possible graphs and the possible graph \(G_1\) (Fig. 1(c)) and \(G_2\) (Fig. 1(d)) have probability \(\Pr[G_1] = 0.000007056\) and \(\Pr[G_2] = 0.00003704\), respectively.

According to the IC model, given a seed node \(s\), the influenceability of \(s\), denoted by \(F_s(\mathcal{G})\), is the expected influence spread over all the possible graphs of \(\mathcal{G}\). Therefore, based on the probabilistic graph model, the influenceability \(F_s(\mathcal{G})\) can be given by

\[
F_s(\mathcal{G}) = \sum_{G_P \in \Omega} \Pr[G_P] f_s(G_P),
\]

where \(\Omega\) denotes the set of all possible graphs of \(\mathcal{G}\), and \(f_s(G_P)\) is the number of nodes that are reachable from the seed node \(s\) in the possible graph \(G_P\). Note that \(f_s(G_P)\) is a random variable and its expectation is \(F_s(\mathcal{G})\), i.e. \(F_s(\mathcal{G}) = \mathbb{E}[f_s(G_P)]\).

As an example, consider the source node \(s = v_5\) in the influence network \(\mathcal{G}\) in Fig. 1(a) \(F_s(\mathcal{G})\) can be computed by enumerating all \(2^{10}\) possible graphs, \(G_P\), and com-
puting the corresponding $\Pr[G_P]$ and $f_s(G_P)$. For instance, from Fig. 1(c) and Fig. 1(d) we have $f_s(G_1) = 3$ and $f_s(G_2) = 5$. In this example, the exact $F_s(G)$ is 0.46123456.

Equipped with the definition of $F_s(G)$, we describe the influenceability evaluation problem as follows.

**Problem Statement:** Given an influence network $G$ and a seed node $s$, the influenceability evaluation problem is to compute the influenceability $F_s(G)$ (Eq. (2)).

It is important to note that the influenceability evaluation problem is known to be $#P$-complete [Chen et al. 2010], even for the very special influence network where the influence probabilities of all edges are equivalent. There is no hope to exactly evaluate the influenceability in polynomial time unless $P = #P$. Given the hardness of this problem, in this paper, our goal is to develop an efficient and accurate approximate algorithm to evaluate $F_s(G)$ given a seed node $s$.

An important metric for evaluating the accuracy of an approximate algorithm is the mean squared error (MSE), which is denoted by $\mathbb{E}[(\hat{F}_s(G) - F_s(G))^2]$, where $\hat{F}_s(G)$ denotes an estimator of $F_s(G)$ by the approximate algorithm. By the so-called variance-bias decomposition [Jin et al. 2011b], this metric can be decomposed into two parts.

$$\mathbb{E}[(\hat{F}_s(G) - F_s(G))^2] = \text{Var}(\hat{F}_s(G)) + \left[\mathbb{E}(\hat{F}_s(G)) - F_s(G)\right]^2,$$

where $\mathbb{E}(\hat{F}_s(G))$ and $\text{Var}(\hat{F}_s(G))$ denote the expectation and variance of the estimator $\hat{F}_s(G)$, respectively. If an estimator is unbiased, then the second term in Eq. (3) will be canceled out. Therefore, the variance of the unbiased estimator becomes the only indicator for evaluating the accuracy of the estimator.
3. NAIVE MONTE-CARLO

In this section, we introduce the Naive Monte-Carlo (NMC) sampling for estimating the influenceability $F_s(G)$ given a seed node $s$, which is the only existing algorithm used in the influence maximization literature [Kempe et al. 2003] [Leskovec et al. 2007] [Chen et al. 2009] [Chen et al. 2010]. This method first samples $N$ possible graphs $G_1, G_2, \cdots, G_N$ of $G$ according to the influence probabilities $P$, and then calculates the number of reachable nodes from the seed node $s$ in each possible graph $G_i, \ i = 1, 2, \cdots, N$, i.e., $f_s(G_i)$. Finally, the NMC estimator $\hat{F}_{NMC}$ is given below.

$$\hat{F}_{NMC} = \frac{\sum_{i=1}^{N} f_s(G_i)}{N}. \quad (4)$$

The NMC estimator is an unbiased estimator of $F_s(G)$, such that $E(\hat{F}_{NMC}) = F_s(G)$. The variance of the NMC estimator is given as follows.

$$\text{Var}(\hat{F}_{NMC}) = \frac{\sum_{i=1}^{N} \text{Pr}[G_i] (f_s(G_i) - \hat{F}_{NMC})^2}{N}. \quad (5)$$

Notice that exactly computing the variance $\text{Var}(\hat{F}_{NMC})$ is extremely expensive, because we have to enumerate all the possible graphs to determine it, whose time complexity is exponential. In practice, we resort to an unbiased estimator of $\text{Var}(\hat{F}_{NMC})$ to evaluate the accuracy of the estimator $\hat{F}_{NMC}$ [Jin et al. 2011b]. In this case, an unbiased estimator of $\text{Var}(\hat{F}_{NMC})$ is given by the following equation.

$$\tilde{\text{Var}}(\hat{F}_{NMC}) = \frac{\sum_{i=1}^{N} (f_s(G_i) - \hat{F}_{NMC})^2}{N - 1} = \frac{\sum_{i=1}^{N} f_s(G_i)^2 - N \hat{F}_{NMC}^2}{N - 1}. \quad (6)$$

According to Eq. (6), $\tilde{\text{Var}}(\hat{F}_{NMC})$ may be very large, because the value of $f_s(G_i)$ falls into the interval $[0, n - 1]$, which may result in $\tilde{\text{Var}}(\hat{F}_{NMC})$ as large as $O(n^2)$. Here, $n$ is the number of nodes in $G$. For example, assume $f_s(G_i) = 0$ for $i = 1, \cdots, N/2$ and $f_s(G_i) = n - 1$ for $i = N/2 + 1, \cdots, N$, then $\tilde{\text{Var}}(\hat{F}_{NMC})$ equals to $N(n - 1)^2/4(N - 1) = O(n^2)$. Therefore, the key issue that we address in this paper is to design more accurate estimators than the NMC estimator for estimating the influenceability $F_s(G)$.

The NMC algorithm is described in Algorithm 1. The algorithm works in $N$ iterations (line 2-5). In each iteration, the NMC algorithm needs to generate a possible graph by tossing $m$ biased coins for $m$ edges in $G$, which takes $O(m)$ time complexity (line 3). Then, the algorithm invokes a BFS algorithm to calculate the number of reachable nodes from $s$, which again causes $O(m)$ time complexity (line 4). As a result, the time complexity of the NMC algorithm is $O(Nm)$.

4. NEW TYPE-I ESTIMATORS

In this section, we first introduce an exact algorithm for computing the influenceability $F_s(G)$, which will guide us to design the new estimators. We will propose two new estimators based on the idea of stratified sampling [Thompson 2002]. Both estimators are shown to be unbiased, and their variance is significantly smaller than the variance of the NMC estimator. We refer to the two estimators as the type-I estimators.
4.1. An exact algorithm

We introduce an exact divide-and-conquer enumeration algorithm to evaluate the influenceability for a given influence network \( G = (V, E, P) \) with \( n \) nodes and \( m \) edges. The main idea of our exact algorithm is described as follows. First, the algorithm divides the entire probability space \( \Omega \) (all the possible graphs) into \( 2^r \) different subspaces by randomly enumerating \( r \) (\( r < m \)) edges that have not been enumerated. Note that \( r \) is a small number (eg. \( r = 5 \)). In each subspace, the exact algorithm recursively enumerates another \( r \) edges, and this process will terminate until all the edges are enumerated. The partition method of the exact algorithm is described in Table I. In Table I, “0”, “1”, and “*” denote the statuses of inexistence, existence, and not-yet enumerated, for the edges, respectively. Each case from 1, 2, \( \cdots \), to \( r \) corresponds to a subspace. And \( \Omega_i \), for \( i = 1, 2, \cdots, 2^r \), denotes the probability space of the case \( i \), which represents the set of all possible graphs in the case \( i \).

To clarify our algorithm, let \( T = (e_1, e_2, \cdots, e_r) \) be the set of selected \( r \) edges, and \( X_i = (X_{i,1}, X_{i,2}, \cdots, X_{i,r}) \) be the status vector corresponding to the selected \( r \) edges under the case \( i \), where \( X_{i,j} = 0 \) signifies that the edge \( e_j \) does not exist, and \( X_{i,j} = 1 \) otherwise. For example, for case 1 in Table I the status vector is \( X_1 = (0, 0, \cdots, 0) \), which means that all the selected \( r \) edges do not exist. In other words, all the possible graphs in \( \Omega_1 \) do not include the edges in \( T \). The probability of a possible graph in case \( i \) is given by

\[
\pi_i = \Pr[\exists P \in \Omega_i] = \prod_{e_j \in T \land X_{i,j} = 1} p_j \prod_{e_j \in T \land X_{i,j} = 0} (1 - p_j). \tag{7}
\]

In addition, let \( A_1 \) be the set of edges that have been enumerated, and \( A_2 \) be the set of edges that have not been enumerated, such that \( A_1 \cup A_2 = E \), and \( A_1 \cap A_2 = \emptyset \). Then, the influenceability of the node \( s \) under the case \( i \) is defined as

\[
F_s(G|A_1, A_2, X_i) = \sum_{G_P \in \Omega_i} f_s(G_P) \frac{\Pr[G_P]}{\pi_i}, \tag{8}
\]

where \( G(A_1, A_2, X_i) \) denotes the set of possible graphs in the case \( i \), i.e. \( \Omega_i \). According to Eq. (8), \( F_s(G(A_1, A_2, X_i)) \) denotes the expected spread over all the possible graphs
The enumeration procedure given in Algorithm 2 can be characterized by a full $2^r$-ary tree which is depicted in Fig. 2. Note that, to simplify our analysis, here we assume that $r$ is divisible by $m$. In the tree, each node represents a probability space that consists of a set of possible graphs. For example, the root node denotes the probability space that includes the set of all possible graphs, and each leaf node denotes the probability space that includes only one possible graph. Each internal node has $2^r$ children, and each child corresponds to a case described in Table 1. To compute $F_s(\mathcal{G})$, we need to traverse all the nodes in the tree. Because the number of nodes in the tree is $O(2^m)$, the time complexity of Algorithm 2 is $O(2^m)$. Therefore, the exact algorithm only works on small networks due to the nature of $\#P$-complete of the influenceability evaluation.
problem. In the following, we will develop two types of efficient approximation algorithms for evaluating the influenceability.

4.2. Basic stratified sampling estimator (I)

As discussed in Section 3, the NMC estimator leads to a large variance. To reduce the variance, we propose a new stratified sampling estimator for influenceability evaluation. We call this new estimator the basic stratified sampling (BSS) estimator, because it serves as the basis for designing recursive stratified sampling (RSS) estimator which will be described in Section 4.3. To distinguish the type-II estimators which will be introduced in Section 5, we refer to the new estimators presented in this section as the type-I estimators. Specifically, we refer to the type-I BSS and RSS estimator as the BSS-I and RSS-I estimator, respectively.

Unlike the NMC sampler which draws a sample (a possible graph) from the entire population (all the possible graphs), the stratified sampling [Thompson 2002] first divides the population into $M$ disjoint groups, which are called strata, and then independently picks separate samples from these groups. Stratified sampling is a commonly used technique for reducing variance [Thompson 2002] in sampling design. There are two key techniques in stratified sampling: stratification, which is a process for partitioning the entire population into disjoint strata, and sample allocation, which is a procedure to determine the sample size that needs to be drawn from each stratum. Below, we will introduce our stratification and sample allocation method.

*Stratification:* Our idea of stratification is based on the exact algorithm described in the previous subsection. First, we choose $r$ edges and determine their statuses ($0/1$), where $r$ is a small number. Recall that this process generates $2^r$ various cases as shown in Table I and thereby it partitions the set of possible graphs $\Omega$ into $2^r$ subsets $\Omega_1, \cdots, \Omega_{2^r}$. Second, we let each subset be a stratum. This is because $\Omega = \bigcup_{i=1}^{2^r} \Omega_i$, thus each case is indeed a valid stratum. It is worth of mentioning that our stratification process corresponds to the top two layers in the enumeration tree (Fig. 2), the root node denotes the entire population, and each child represents a stratum. The stratification process is depicted in Table I.
In our stratification approach, a question that arises is how to select the $r$ edges for stratification. As shown in our experiments, the edge-selection strategy for choosing $r$ edges significantly affects the performance of the estimator. One straightforward strategy is to randomly pick $r$ edges from the edge set $E$. We refer to this edge selection strategy as the random edge-selection strategy. With this strategy, the selected $r$ edges may not have direct contributions for computing the influenceability. For example, in Fig. 1(b) for the source node $s = v_5$, assume $r = 2$ and the selected edges are $\{v_1 \rightarrow v_2, v_6 \rightarrow v_2\}$. The edges $\{v_1 \rightarrow v_2, v_6 \rightarrow v_2\}$ have no direct contributions for calculating the influenceability $F_s(G)$. This may reduce the performance of the BSS-I estimator. For avoiding such a problem, we introduce another heuristic edge-selection strategy based on the BFS visiting order of the edges. To estimate $F_s(G)$, we first perform a BFS algorithm starting from the node $s$ to obtain the first $r$ edges according to the BFS visiting order of the edges. Then, we use these $r$ edges for stratification. We refer to such edge-selection strategy as the BFS edge-selection strategy. Consider the same example in Fig. 1(b), assume $r = 2$, the first $r$ edges are $\{v_5 \rightarrow v_3, v_5 \rightarrow v_2\}$. Then, we partition the population into 4 strata according to the statuses of these two edges. Obviously, according to the BFS edge-selection strategy, the selected edges have direct contribution to calculate the influenceability. In our experiments, we find that the performance of the BSS-I estimator with BFS edge-selection strategy is significantly better than the performance of the BSS-I estimator with random edge-selection strategy.

The BSS-I estimator: Let $N$ be the total number of samples, $N_i$ be the number of samples drawn from the stratum $i$ ($i = 1, 2, \cdots, 2^r$), and $G_{i,j}$ ($j = 1, 2, \cdots, N_i$) be a possible graph sampled from the stratum $i$. Then, the BSS-I estimator is given as follows.

$$\hat{F}_{BSSI} = \sum_{i=1}^{2^r} \frac{1}{N_i} \sum_{j=1}^{N_i} f_s(G_{i,j}),$$

where $\pi_i$ is defined in Eq. (7). The following theorem shows that $\hat{F}_{BSSI}$ is an unbiased estimator of the influenceability $F_s(G)$.

**Theorem 4.2.** $F_s(G) = \mathbb{E}(\hat{F}_{BSSI})$.

**Proof.** We prove it by the following equalities.

$$\mathbb{E}(\hat{F}_{BSSI}) = \mathbb{E}\left(\sum_{i=1}^{2^r} \frac{1}{N_i} \sum_{j=1}^{N_i} f_s(G_{i,j})\right)
= \sum_{i=1}^{2^r} \frac{1}{N_i} \mathbb{E}(f_s(G_{i,j}))
= \sum_{i=1}^{2^r} \pi_i \sum_{G_P \in \Omega_i} f_s(G_P) \frac{P[G_P]}{\pi_i}
= \sum_{G_P \in \Omega} \Pr[G_P] f_s(G_P)
= F_s(G)$$

Let $\sigma_i$ be the variance of the sample in the stratum $i$. Since the samples are independently drawn by the basic stratified sampling algorithm, thus the variance of the
BSS-I estimator is given by

\[ \text{Var}(\hat{F}_{\text{BSS-I}}) = \sum_{i=1}^{2^r} \pi_i^2 \frac{\sigma_i}{N_i}, \]

where \( \pi_i \) is given in Eq. (7).

**Sample allocation**: As discussed above, the BSS-I estimator is unbiased and the variance of the BSS-I estimator depends on the sample size of all the strata, i.e., \( N_i \), for \( i = 1, 2, \ldots, 2^r \). Thus, a question that arises is how to allocate the sample size for each stratum \( i (i = 1, 2, \ldots, 2^r) \) to minimize the variance of the BSS-I estimator, i.e. \( \text{Var}(\hat{F}_{\text{BSS-I}}) \). Formally, the sample allocation problem is formulated as follows.

\[
\min \ \text{Var}(\hat{F}_{\text{BSS-I}}) = \sum_{i=1}^{2^r} \pi_i^2 \frac{\sigma_i}{N_i}, \\
\text{s.t.} \quad \sum_{i=1}^{2^r} N_i = N. \tag{12}
\]

By applying the Lagrangian method, we can derive the optimal sample allocation as given by

\[ N_i = N\pi_i \sqrt{\frac{\sigma_i}{\sum_{i=1}^{2^r} \pi_i \sqrt{\sigma_i}}}, \tag{13} \]

for \( i = 1, \ldots, 2^r \). From Eq. (13), the optimal allocation needs to know the variance of the sample in each stratum, i.e. \( \sigma_i \), for \( i = 1, \ldots, 2^r \). However, such variances are unavailable in our problem. Interestingly, we find that, if the sample size of the stratum \( i \) is allocated to \( \pi_i N \), then the variance of the BSS-I estimator will be smaller than the variance of the NMC estimator. We have the following theorem.

**Theorem 4.3.** If \( N_i = \pi_i N \), then \( \text{Var}(\hat{F}_{\text{BSS-I}}) \leq \text{Var}(\hat{F}_{\text{NMC}}) \).

**Proof.** If \( N_i = \pi_i N \), then we have \( \text{Var}(\hat{F}_{\text{BSS-I}}) = \sum_{i=1}^{2^r} \pi_i \frac{\sigma_i}{N_i} \). Let \( \mu_i = \mathbb{E}(f_s(G_{i,j})) \) be the expectation of the sample in the stratum \( i \). By definition, we have \( \sigma_i = \mathbb{E}(f_s(G_{i,j})^2) - \mu_i^2 = \sum_{G_P \in \Omega_i} f_s(G_P)^2 \frac{\text{Pr}[G_P]}{\pi_i} - \mu_i^2 \). Then, we have

\[ \text{Var}(\hat{F}_{\text{BSS-I}}) = \frac{1}{N} \sum_{i=1}^{2^r} \pi_i (\sum_{G_P \in \Omega_i} f_s(G_P)^2 \frac{\text{Pr}[G_P]}{\pi_i} - \mu_i^2) \]

\[ = \frac{1}{N} \sum_{i=1}^{2^r} (\sum_{G_P \in \Omega_i} f_s(G_P)^2 \text{Pr}[G_P] - \pi_i \mu_i^2) \]

\[ = \frac{1}{N} \sum_{G_P \in \Omega_i} \text{Pr}[G_P] f_s(G_P)^2 - \frac{1}{N} \sum_{i=1}^{2^r} \pi_i \mu_i^2. \]

Given this, we can derive the difference between \( \text{Var}(\hat{F}_{\text{BSS-I}}) \) and \( \text{Var}(\hat{F}_{\text{NMC}}) \) (Eq. (5)) as follows:

\[ \text{Var}(\hat{F}_{\text{NMC}}) - \text{Var}(\hat{F}_{\text{BSS-I}}) \]

\[ = \frac{1}{N} \sum_{i=1}^{2^r} \pi_i \mu_i^2 - (\mathbb{E}[f_s(G_P)])^2 \]

\[ = \frac{1}{N} \sum_{i=1}^{2^r} \pi_i \mu_i^2 - (\sum_{G_P \in \Omega} \text{Pr}[G_P] f_s(G_P))^2 \]

\[ = \frac{1}{N} \sum_{i=1}^{2^r} \pi_i \mu_i^2 - (\sum_{i=1}^{2^r} \pi_i \mu_i^2)^2 \]

\[ = \frac{\text{Var}(\mu_i)}{N} \geq 0. \]

Note that in the last equality \( \mu_i \) can be treated as a random variable. Then, we have \( \sum_{i=1}^{2^r} \pi_i \mu_i^2 = \mathbb{E}(\mu_i^2) \) and \( (\mathbb{E}(\mu_i))^2 = (\sum_{i=1}^{2^r} \pi_i \mu_i^2)^2 \), thus the last equality holds. This completes the proof. \( \square \)
Algorithm 3: BSS-I (G, N, s)

Input: Influence network G, sample size N, and the seed node s.
Output: The BSS-I estimator $\hat{F}$.

1: $\hat{F} \leftarrow 0$;
2: Choose $r$ edges according to an edge-selection strategy;
3: for $i = 1$ to $2^r$ do
4: Let $X_i$ be the status vector of stratum $i$;
5: Compute $\pi_i$ by Eq. (7);
6: $N_i \leftarrow \lceil \pi_i N \rceil$;
7: $t \leftarrow 0$;
8: for $j = 1$ to $N_i$ do
9: Flip $m - r$ coins to determine the rest $m - r$ edges;
10: Let $Y_j$ be the status vector of the rest $m - r$ edges;
11: Append $X_i$ to $Y_j$ to generate a possible graph $G_j$;
12: Compute $f_s(G_j)$ by the BFS algorithm;
13: $t \leftarrow t + f_s(G_j)$;
14: $t \leftarrow t/N_i$;
15: $\hat{F} \leftarrow \hat{F} + \pi_i t$;
16: return $\hat{F}$;

The BSS-I algorithm: Given the stratification and sample allocation methods, we present our basic stratified sampling algorithm in Algorithm 3. First, Algorithm 3 selects $r$ edges to partition the population into $2^r$ strata according to an edge-selection strategy (line 2), either random or BFS edge-selection. For convenience, we refer to the BSS-I estimator with random edge-selection and the BSS-I estimator with BFS edge-selection as BSS-I-RM and BSS-I-BFS estimator, respectively. Second, according to our sample allocation method, the algorithm draws $\pi_i N$ samples from the stratum $i$ (line 8-13). Finally, the algorithm outputs the BSS-I estimator $\hat{F}_{BSSI}$. Notice that it takes $O(m)$ time for both generating a possible graph $G$ and performing BFS on $G$. Besides, the algorithm needs to draw $N$ possible graphs. Hence, the time complexity of Algorithm 3 is $O(mN)$, which has the same complexity as the NMC estimator. However, our BSS-I estimator significantly reduces the variance of the NMC estimator. The advantages of the BSS-I estimator are twofold. On one hand, given the sample size, the BSS-I estimator is more accurate than the NMC estimator as it has a smaller variance. On the other hand, to achieve the same variance, the BSS-I estimator needs a smaller sample size than that of the NMC estimator, thus it reduces the time complexity of the sampling process.

4.3. Recursive stratified sampling estimator (I)

Recall that the BSS-I estimator splits the entire set of possible graphs into $2^r$ subsets, which corresponds to the top two layers in the enumeration tree (Fig. 2). Interestingly, we observe that the basic stratified sampling (BSS-I) can be applied into any internal nodes of the enumeration tree. Based on this observation, we develop a recursive stratified sampling estimator, namely RSS-I estimator, which is described in Algorithm 4. The RSS-I estimator recursively partitions the sample size $N$ to $N_i = \pi_i N$ ($i = 1, 2, \cdots, 2^r$) for estimating the influenceability at the stratum $i$ (line 9-19). Note that since the BSS-I estimator is unbiased, the RSS-I estimator is also unbiased. Moreover, RSS-I reduces the variance at each partition, thus the variance of RSS-I is significantly smaller than the variance of BSS-I as stated by the following theorem.
Theorem 4.4. Let $\text{Var}(\hat{F}_{RSS-I})$ be the variance of RSS-I, then $\text{Var}(\hat{F}_{RSS-I}) \leq \text{Var}(\hat{F}_{RSS-II})$.

Proof. We focus on the case that RSS-I only partitions the population $2^r + 1$ times. Similar arguments can be used to prove the case of more partitions. At the first partition, RSS-I splits the population into $2^r$ strata, which is equivalent to BSS-I. In each stratum $i$ ($i = 1, \cdots, 2^r$), RSS-I recursively partitions it into $2^r$ sub-strata. Let $\Omega_i$, $\mu_i$, $\sigma_i$ and $N_i$ be the probability space, the expectation, the variance, and the sample size of the stratum $i$ at the first partition, respectively. Let $\pi_i = \Pr\{G_P \in \Omega_i\}$ be the probability of a sample in stratum $i$ as defined in Eq. (7). Similarly, for each stratum $i$, we denote the probability space, the expectation, the variance, and the sample size of the sub-stratum $k$ ($k = 1, \cdots, 2^r$), as $\Omega_{i,k}$, $\mu_{i,k}$, $\sigma_{i,k}$, and $N_{i,k}$, respectively. Further, we denote the probability of a sample in a sub-stratum $k$ as $\pi_{i,k}$, i.e., $\pi_{i,k} = \Pr\{G_P \in \Omega_{i,k}\}$.

Then, we have $\pi_{i,k} = \pi_i \omega_k$, where $\omega_k$ denotes the probability of a sample in sub-stratum $k$ conditioning on it is in stratum $i$, i.e., $\omega_k = \Pr\{G_P \in \Omega_{i,k} | G_P \in \Omega_i\}$.

The RSS-I estimator is given by $\hat{F}_{RSS-I} = \sum_{i=1}^{2^r} \sum_{k=1}^{2^r} \pi_{i,k} \frac{1}{N_{i,k}} \sum_{j=1}^{N_{i,k}} f_s(G_{i,k,j})$, where $G_{i,k,j}$ ($j = 1, \cdots, N_{i,k}$) denotes a possible graph sampled from the sub-stratum $k$ of the stratum $i$. Then, the variance of RSS-I is $\text{Var}(\hat{F}_{RSS-I}) = \sum_{i=1}^{2^r} \sum_{k=1}^{2^r} \frac{\pi_{i,k}^2 \sigma_{i,k}}{N_{i,k}}$. By our sample allocation strategy, we have $N_i = N_{i,k}$, thereby the variance can be simplified to $\text{Var}(\hat{F}_{RSS-I}) = \sum_{i=1}^{2^r} \frac{1}{N_i} \sum_{k=1}^{2^r} \pi_{i,k} \sigma_{i,k}$. Further, by $\pi_{i,k} = \pi_i \omega_k$, we have $\text{Var}(\hat{F}_{RSS-I}) = \sum_{i=1}^{2^r} \frac{1}{N_i} \sum_{k=1}^{2^r} \omega_k \sigma_{i,k}$. By the proportional sample allocation, we have $\text{Var}(\hat{F}_{RSS-I}) = \sum_{i=1}^{2^r} \frac{\sigma_i}{N_i}$.

Therefore, the proof is completed followed by $\sum_{k=1}^{2^r} \omega_k \sigma_{i,k} \leq \sigma_i$. By definition, we have

$$
\sum_{k=1}^{2^r} \omega_k \sigma_{i,k} = \sum_{k=1}^{2^r} \omega_k \left( \mathbb{E}(f_s(G_{i,k,j}))^2 - \mu_{i,k}^2 \right)
= \sum_{k=1}^{2^r} \omega_k \left( \sum_{G_P \in \Omega_{i,k}} \frac{\Pr\{G_P\}}{\pi_{i,k}} f_s(G_P)^2 - \mu_{i,k}^2 \right)
= \sum_{k=1}^{2^r} \sum_{G_P \in \Omega_{i,k}} \frac{\Pr\{G_P\}}{\pi_i} f_s(G_P)^2 - \sum_{k=1}^{2^r} \omega_k \mu_{i,k}^2
= \sum_{G_P \in \Omega_i} \frac{\Pr\{G_P\}}{\pi_i} f_s(G_P)^2 - \sum_{k=1}^{2^r} \omega_k \mu_{i,k}^2.
$$

Then, we have

$$
\sigma_i - \sum_{k=1}^{2^r} \omega_k \sigma_{i,k} = \sum_{k=1}^{2^r} \omega_k \mu_{i,k}^2 - \mu_i^2 = \sum_{k=1}^{2^r} \omega_k \mu_{i,k}^2 - (\sum_{k=1}^{2^r} \omega_k \mu_{i,k})^2 \geq 0.
$$

This completes the proof. □

The RSS-I algorithm terminates until the sample size becomes smaller than a given threshold ($r$) or the number of unsampled edges smaller than $r$ (line 2). When the terminative conditions of the RSS-I algorithm satisfy, we perform a naive Monte-Carlo sampling for estimating the influenceability (line 3-7).

Similar to the BSS-I estimator, the partition approach in RSS-I estimator also depends on the edge-selection strategy (line 9). Likewise, we have two edge-selection strategies for the RSS-I estimator, either random edge-selection or BFS edge-selection. For convenience, we refer to the RSS-I estimator with random edge-selection and with BFS edge-selection as the RSS-I-RM and RSS-I-BFS estimator, respectively.

Reconsider the example in Fig. 1(b), the BFS visiting order of the edges is $\{v_5 \to v_3, v_5 \to v_6, v_3 \to v_1, v_3 \to v_4, v_6 \to v_2, v_1 \to v_2, v_1 \to v_3, v_1 \to v_4, v_4 \to v_6, v_2 \to v_6\}$. Assume $r = 2$, according to the BFS visiting order, then the RSS-I-BFS first picks edge $v_5 \to v_3$ and $v_5 \to v_6$ for stratification, and then selects the edges $v_3 \to v_1$ and $v_3 \to v_4$, and so on. It worth of mentioning that we can invoke the procedure RSS-I ($G, \emptyset, E, \emptyset, N, s$), where $s$ is the seed node, to calculate the RSS-I estimator.
1. \( \hat{F} \leftarrow 0 \);
2. if \( N < \tau \) or \( |E_2| < r \) then
3. for \( j = 1 \) to \( N \) do
4. Flip \( |E_2| \) coins to generate a possible graph \( G_j \);
5. Compute \( f_s(G_j) \) by the BFS algorithm;
6. \( \hat{F} \leftarrow \hat{F} + f_s(G_j) \);
7. return \( \frac{\hat{F}}{N} \);
8. else
9. Select \( r \) edges from \( E_2 \) according to an edge-selection strategy \{Random or BFS visiting order\};
10. Let \( T \) be the set of selected edges;
11. for \( i = 1 \) to \( 2^r \) do
12. \( Y \leftarrow X \) \{Recording the current status vector \( X \)\};
13. Let \( X_i \) be the status vector of set \( T \) in stratum \( i \);
14. Append \( X_i \) to \( Y \);
15. Compute \( \pi_i \) by Eq. (7);
16. \( N_i \leftarrow \lfloor \pi_i N \rfloor \);
17. \( \mu_i \leftarrow \text{RSS-I}(G, E_1 \cup T, E_2 \setminus T, Y, N_i, s) \);
18. \( \hat{F} \leftarrow \hat{F} + \pi_i \mu_i \);
19. return \( \hat{F} \);

We analyze the time complexity of Algorithm 4. For sampling a possible graph, Algorithm 4 needs to traverse the enumeration tree (Fig. 2) from the root node to the terminative node. Here the terminative node is a node in the enumeration tree where the terminative conditions of the recursion satisfy at that node, i.e. \( N < \tau \) or \( |E_2| < r \) holds in Algorithm 4. Let \( d \) be the average length of the path from the root node to the terminative node. Then, by analysis, the time complexity of the algorithm at each internal node of the path is \( O(r) \). Suppose that the total number of such paths is \( K \). Then, the algorithm takes \( O(Kd) \) time complexity at the internal nodes of all the paths. Note that \( K \) is bounded by the sample size \( N \), and \( d \) is a very small number w.r.t. \( N \). More specifically, we can derive that \( d = O(\log_2 N) \), which is a very small number. For example, assume \( r = 5 \) and \( N = 100,000 \), then we can get \( d \approx 3.3 \). For all the terminative nodes, the time complexity of the algorithm is \( O(Nm) \). This is because the algorithm needs to sample \( N \) possible graphs in total over all the terminative nodes, and for each possible graph the algorithm performs a BFS to compute the influenceability which takes \( O(m) \) time complexity. Since \( O(Kd) \) is dominated by \( O(Nm) \), the time complexity of Algorithm 4 is \( O(Nm + Kd) = O(Nm) \).

5. NEW TYPE-II ESTIMATORS
In this section, we propose two new stratified sampling estimators, namely type-II basic stratified sampling (BSS-II) estimator and type-II recursive stratified sampling (RSS-II) estimator. The BSS-II and RSS-II are shown to be unbiased and their variance are significantly smaller than the variance of the NMC estimator. In the following, we first introduce the BSS-II estimator, and then present the RSS-II estimator.
Table III. Stratum design of the BSS-II/RSS-II estimator

| Edges | e₁ e₂ e₃ · · · eᵣ eᵣ+1 · · · eₑm | Prob. space |
|-------|---------------------------------|-------------|
| Stratum 0 | 0 0 0 · · · 0 * · · · * | Ω₀ |
| Stratum 1 | 1 * · · · * · · · * | Ω₁ |
| Stratum 2 | 0 1 * · · · * · · · * | Ω₂ |
| Stratum 3 | 0 0 1 · · · * · · · * | Ω₃ |
| ... | ... | ... |
| Stratum r | 0 0 0 · · · 1 * · · · * | Ωᵣ |

5.1. Basic stratified sampling estimator (II)

**Stratification:** We propose a new stratification method for the BSS-II estimator. This new stratification method splits the entire probability space Ω into r + 1 various subspaces (Ω₀, · · · , Ωᵣ) by choosing r edges. Specifically, for stratum 0, we set the statuses of all the r selected edges to “0”, and for the stratum i (i ≠ 0), we set the status of edge i to “1” and the statuses of all the previous i − 1 edges (i.e. e₁, · · · , eᵢ−1) to “0”. Unlike the stratification method of the BSS-I estimator, this new stratification approach allows us to set r to be a big number, such as r = 50. The stratum design method is depicted in Table III.

In Table III, each stratum (Stratum 0, Stratum 1, · · · , Stratum r) corresponds to a subspace (Ω₀, Ω₁, · · · , Ωᵣ). For any i ≠ j, we have Ωᵢ ∩ Ωⱼ = ∅. Below, we show that ∪ᵢ=₀⁻¹Ωᵢ = Ω. Let T = (e₁, e₂, · · · , eᵣ) be the set of r selected edges and pᵢ (i = 1) be the corresponding influence probability, then the probability of a possible graph in stratum i is given by

\[
π'_i = \Pr[G_P ∈ Ω_i] = \begin{cases} 
    \prod_{j=1}^{r} (1 - p_j), & \text{if } i = 0 \\
    p_i \prod_{j=1}^{r} (1 - p_j), & \text{otherwise}
\end{cases}
\]  

(14)

The following theorem implies ∪ᵢ=₀⁻¹Ωᵢ = Ω.

**Theorem 5.1.** \( \Pr[G_P ∈ Ω] = \sum_{i=0}^{r} \Pr[G_P ∈ Ω_i] = 1. \)

**Proof.** We prove it by the following equalities.

\[
\begin{align*}
\sum_{i=0}^{r} \Pr[G_P ∈ Ω_i] &= \prod_{j=1}^{r} (1 - p_j) + p_1 (1 - p_1) p_2 + \cdots + p_r \prod_{j=1}^{r-1} (1 - p_j) \\
&= \prod_{j=1}^{r} (1 - p_j) + p_1 (1 - p_1) p_2 + \cdots + p_r \prod_{j=1}^{r-1} (1 - p_j) \\
&= 1 - p_1 + p_1 \\
&= 1
\end{align*}
\]

Armed with Theorem 5.1, we conclude that the stratum design approach described in Table III is a valid stratification method.

The **BSS-II estimator**: Similar to the BSS-I estimator, we let N be the total sample size, and Nᵢ be the sample size of the stratum i, and Gᵢ,j (j = 1, 2, · · · , Nᵢ) be a possible graph sampled from the stratum i. Then the BSS-II estimator \( \hat{F}_{BSSII} \) is given by

\[
\hat{F}_{BSSII} = \sum_{i=0}^{r} \frac{π'_i}{N_i} \sum_{j=1}^{N_i} f_s(G_{i,j}),
\]

(15)

where π’ᵢ is given in Eq. (14). Similar to Theorem 4.2, the following theorem shows that the BSS-II estimator is unbiased. The proof is similar to the proof of Theorem 4.2 thus we omit for brevity.
The variance of the BSS-II estimator is given by

\[ \text{Var}(\hat{\mu}_{BSSII}) = \sum_{i=0}^{r} \pi_i^2 \sigma_i^2 / N_i, \]  

(16)

where \( \sigma_i \) denotes the variance of the sample in the stratum \( i \).

**Sample allocation:** Analogous to the BSS-I estimator, for the BSS-II estimator, we can derive that the optimal sample allocation is given by 
\[ N_i = n_i' \pi'_i \sqrt{\sigma_i} / \sum_{i=0}^{r} \pi'_i \sqrt{\sigma_i}. \]

This optimal allocation strategy needs to know the variance of the sample in each stratum, which is impossible in our problem. Therefore, similar to the sample allocation approach used in the BSS-I estimator, for the BSS-II estimator, we set the sample size of the stratum \( i \) equals to \( \pi'_i N \), i.e. \( N_i = \pi'_i N \). On the basis of this sample allocation method, we show that the variance of the BSS-II estimator is smaller than the variance of the NMC estimator as stated by the following theorem. The proof of the theorem is similar to theorem [4.3], thus we omitted for brevity.

**Theorem 5.3.** If \( N_i = \pi'_i N \), \( \text{Var}(\hat{\mu}_{BSSII}) \leq \text{Var}(\hat{\mu}_{NMC}) \).

However, it is very hard to compare the variance of the BSS-II estimator with the variance of the BSS-I estimator. In our experiments, we find that these two estimators achieve comparable variance.

**The BSS-II algorithm:** With the stratification and sample allocation method, we describe the BSS-II algorithm in Algorithm 5. Algorithm 5 picks \( r \) edges to split the entire population into \( r + 1 \) strata in terms of an edge-selection strategy (line 2). Any of the two edge-selection strategies (random edge-selection and BFS edge-selection) used in the BSS-I algorithm can also be used in the BSS-II algorithm. We refer to the BSS-II estimator with the random edge-selection and the BSS-II estimator with BFS edge-selection as BSS-II-RM and BSS-II-BFS estimator, respectively. In terms of the sample allocation method of the BSS-II estimator, Algorithm 5 picks \( N_i = \pi'_i N \) samples from the stratum \( i \), for \( i = 0, 1, \ldots, r \), and outputs the BSS-II estimator \( \hat{\mu}_{BSSII} \). Like the BSS-I estimator, the time complexity of BSS-II estimator is \( O(Nm) \). This is because the BSS-II needs to draw \( N \) possible graphs, and both sampling each possible graph \( G \) and computing \( F_s(G) \) take \( O(m) \) time.

### 5.2. Recursive stratified sampling estimator (II)

Based on the BSS-II estimator, in this subsection, we develop another new recursive stratified sampling estimator, namely RSS-II estimator. Similar to the idea of the RSS-I estimator, the RSS-II estimator makes use of the BSS-II estimator as the basic component and recursively applies the BSS-II estimator at each stratum. More specifically, the RSS-II estimator first partitions the entire probability space \( \Omega \) into \( r + 1 \) subspace \( \Omega_i \) \( (i = 0, 1, \ldots, r) \) according to the stratification method of the BSS-II estimator. The same partition procedure is recursively performed in each subspace \( \Omega_i \). At each partition, the RSS-II estimator utilizes the same sample allocation method as the BSS-II estimator to allocate the sample size. The recursion process of the RSS-II estimator will terminate until the sample size is smaller than a given threshold \( (\tau) \) or the number of unsampled edges is smaller than \( r \). Since the BSS-II estimator is unbiased, the RSS-II estimator is also unbiased. The variance of the RSS-II estimator is smaller than the variance of the BSS-II estimator, because the RSS-II estimator recursively reduces variance at each partition while the BSS-II estimator only reduces variance at one partition. Similar to Theorem [4.4], we have the following theorem.
mator (Table III). For example, at the first partition of the RSS-II the consume $O$ is $X$ node the algorithm only needs to select $r$ edges and determine their statuses which is denoted by $E_i$ for $i = 0, \ldots, r$ (line 9). In line 15 and line 20, we let $X_i$ be the status vector of the selected edges under the stratum $i$. Unlike the RSS-I estimator, the status vector of the RSS-II estimator is determined by the stratification method of the RSS-II estimator (Table III). For example, at the first partition of the RSS-II estimator, assume $T = (e_1, e_2, \ldots, e_r)$ is the set of $r$ edges selected, the status vector of these selected edges under the stratum 0 is $X_0 = (0, 0, \ldots, 0)$. The status vector under the stratum $i$ is $X_i = (0, \ldots, 0, 1, s, \ldots, *)$, where the statuses of the first $i - 1$ edges are “0”, the status of the $i$-th edge is “1”, and the rest $r - i$ edges are “*”. Finally, the algorithm outputs the RSS-II estimator (line 24).

Like the RSS-I estimator, to sample a possible graph, the RSS-II algorithm needs to traverse the recursive tree from the root node to the terminative node. At all the terminative nodes, the algorithm needs to sample $N$ possible graphs in total, and for each graph it needs to perform a BFS to compute the influenceability, thus the time complexity is $O(Nm)$. At each internal node in a path from the root node to the terminative node, the time complexity is $O(r)$. This is because at each internal node the algorithm only needs to select $r$ edges and determine their statuses which consume $O(r)$ time complexity. Let $d$ be the average length of such path and $K$ be the total number of paths. Then, for all the internal nodes, the algorithm takes $O(Kdr)$
Algorithm 6: RSS-II \((G, E_1, E_2, X, N, s)\)

**Input:** Influence network \(G\), the set of sampled edges \(E_1\), the set of unsampled edges \(E_2\), sample size \(N\), and the seed node \(s\).

**Output:** The RSS-II estimator \(\hat{F}\).

1: \(\hat{F} \leftarrow 0\);
2: if \(N < \tau\) or \(|E_2| < r\) then
3:     for \(j = 1\) to \(N\) do
4:         Flip \(|E_2|\) coins to generate a possible graph \(G_j\);
5:         Compute \(f_s(G_j)\) by the BFS algorithm;
6:         \(\hat{F} \leftarrow \hat{F} + f_s(G_j)\);
7:     return \(\hat{F} / N\);
8: else
9:     Select \(r\) edges from \(E_2\) according to an edge-selection strategy (random or BFS visiting order);
10:     Let \(T = (e_1, e_2, \cdots, e_r)\) be the set of selected edges;
11:     for \(i = 0\) to \(r\) do
12:         Compute \(\pi_i'\) by Eq. (14);
13:         \(N_i \leftarrow \lfloor \pi_i' N \rfloor\);
14:         if \(i = 0\) then
15:             Let \(X_0\) be the status vector of set \(T\) under stratum 0;
16:             Append \(X_0\) to \(X\);
17:             \(\mu_0 \leftarrow \text{RSS-II}(G, E_1 \cup T, E_2 \setminus T, X, N_i, s)\);
18:         else
19:             Let \(T_i = \{e_1, \cdots, e_i\}\);
20:             Let \(X_i\) be the status vector of set \(T_i\) under stratum \(i\);
21:             Append \(X_i\) to \(X\);
22:             \(\mu_i \leftarrow \text{RSS-II}(G, E_1 \cup T_i, E_2 \setminus T_i, X, N_i, s)\);
23:         \(\hat{F} \leftarrow \hat{F} + \pi_i' \mu_i\);
24:     return \(\hat{F}\);

The time complexity. According to the terminative condition given in Algorithm 6, we can derive that \(\bar{d} = \min\{\log_r N, \log_r m\}\). Since \(r\) can be a big number (e.g. \(r = 50\)), \(\bar{d}\) is very small. Thus, the time complexity at the internal nodes \(O(K \bar{d} r)\) can be dominated by \(O(N m)\). We conclude that the average time complexity of Algorithm 6 is \(O(N m)\).

6. EXPERIMENTS

We conduct experimental studies for different estimators over four datasets. We confirm the efficiency and accuracy of the proposed estimators. In the following, we first describe the experimental setup, and then report our results.

6.1. Experimental setup

**Datasets:** We use one synthetic dataset and three real datasets in our experiments. We apply the same parameters used in Jin et al. 2011b to generate the synthetic dataset. For the graph topology, we generate an Erdos-Renyi (ER) random graph with 5,000 vertices and edge density 10. For the influence probabilities, we generate a probability for each edge according to a \([0,1]\) uniform distribution.

The three real datasets are given as follows. (1) FacebookLike dataset: this dataset originates from a Facebook social network for students at University of California, Irvine. It contains the users who sent or received at least one message. We collect this dataset from [toreopsahl.com/datasets](http://toreopsahl.com/datasets). The dataset is a weighted graph, and
Table IV. Summary of the datasets

| Name       | Nodes | Edges   | Ref.                        |
|------------|-------|---------|-----------------------------|
| Random graph | 5,000 | 50,616  | [Jin et al. 2011b]          |
| FacebookLike| 1,899 | 20,296  | [Opsahl and Panzarasa 2009] |
| Condmat    | 16,264| 95,188  | [Newman 2001]              |
| DBLP       | 78,648| 376,515 | [Zhou et al. 2010]         |

the weight of each edge denotes the number of messages passing over the edge. (2) Condmat dataset: this dataset is a weighted collaboration network, where the weight of an edge represents the number of co-authored papers between two collaborators. We download this dataset from [www-personal.umich.edu/~mejn/netdata]. (3) DBLP dataset: this dataset is also a weighted collaboration network, where the weight of the edge signifies the number of co-authored papers. This dataset is provided by the authors in [Zhou et al. 2010]. Table IV summarizes the information for the four real datasets. To obtain the influence networks, for each real dataset, we generate the influence probabilities according to the same method used in [Potamias et al. 2010; Jin et al. 2011b]. Specifically, to generate the probability of an edge, we apply an exponential cumulative distribution function (CDF) with mean 2 to the weight of the edge.

**Different estimators:** In our experiments, we compare 10 estimators. (1) The NMC estimator, which is the Naive Monte-Carlo estimator. (2) RSS-I-RM \( (r = 1) \), which is a special RSS-I-RM estimator where the parameter \( r = 1 \), based on work presented in [Jin et al. 2011b] for computing distance-constraint reachability on uncertain graph. We also generalize their estimator to arbitrary parameter \( r \), and apply the generalized estimator for influenceability evaluation. Recall that beyond the random edge-selection strategy, we propose a more accurate RSS-I estimator with BFS edge-selection strategy. (3) BSS-I, which is the BSS-I estimator with the random edge-selection. (4) BSS-I-BFS, which is the BSS-I estimator with the BFS edge-selection. (5) RSS-I-RM, which is the RSS-I estimator with the random edge-selection. (6) RSS-I-BFS, which is the RSS-I estimator with the BFS edge-selection. (7) BSS-II-RM, which is the BSS-II estimator with the random edge-selection. (8) BSS-II-BFS, which is the BSS-II estimator with the BFS edge-selection. (9) RSS-II-RM, which is the RSS-II estimator with the random edge-selection. (10) RSS-II-BFS, which is the RSS-II estimator with the BFS edge-selection.

**Evaluation metric:** Two metrics are used to evaluate the performance of the estimators: running time and relative variance. The running time evaluates the efficiency of the estimators. The relative variance is leveraged to evaluate the accuracy of the estimators. Let \( \sigma_{NMC} \) be the variance of the NMC estimator. We calculate the relative variance of an estimator \( \hat{F} \) by \( \sigma_{\hat{F}}/\sigma_{NMC} \). Since computing the exact variance of the estimators is intractable, we resort to an unbiased estimator of the variance. Similar evaluation metric has been used in [Jin et al. 2011b]. Specifically, for a given seed node \( s \) in our experiments, we run all the estimators \( \hat{F}_s(G) \) 500 times, thereby we can obtain 500 estimating results: \( \hat{F}_s^{(1)}(G), \hat{F}_s^{(2)}(G), \ldots, \hat{F}_s^{(500)}(G) \). An unbiased variance estimator of \( \hat{F}_s(G) \) is given by

\[
\sum_{i=1}^{500} (\hat{F}_s^{(i)}(G) - \hat{F}_s(G))^2 / 499,
\]

where \( \hat{F}_s(G) \) denotes the mean of the 500 various estimating results.

**Parameter settings and the experimental environment:** Without specifically stated, in all of our experiments, we set the parameters as follows. For all estimators, we set the sample size \( N = 1,000 \). For the BSS-I and RSS-I estimators, we set \( r = 5, \)
and for the BSS-II and RSS-II estimators, we set \( r = 50 \). For the threshold parameter \( \tau \) in Algorithm 4 and Algorithm 6, we set \( \tau = 10 \). All the experiments are conducted on the Scientific Linux 6.0 workstation with 2xQuad-Core Intel(R) 2.66 GHz CPU, and 4G memory. All algorithms are implemented by GCC 4.4.4.

### 6.2. Experimental Results

For all the experiments, we randomly generate 1,000 seed nodes, and the results are the average result over all the seeds. We report our experimental results on random graph, FacebookLike, Condmat, and DBLP dataset in Table V, Table VI, Table VII, and Table VIII, respectively.

From Table V among all the estimators, we can observe that the RSS-I-BFS is the winner on the random graph dataset, the RSS-I-RM, RSS-II-RM, and RSS-II-BFS estimators are significantly better than the RSS-I-RM (\( r = 1 \)) estimator. The specific RSS-I-RM (\( r = 1 \)) estimator outperforms the BSS estimators, and all the BSS estimators are better than the NMC estimator. In particular, RSS-I-BFS reduces the relative variance over the NMC and RSS-I-RM (\( r = 1 \)) estimators by 386% and 227%, respectively. RSS-II-BFS cuts the relative variance over NMC and RSS-I-RM (\( r = 1 \)) by 385% and 226%, respectively. Both RSS-I-RM and RSS-II-RM estimators cut the relative variance over the NMC and the RSS-I-RM (\( r = 1 \)) estimators more than 185% and 91.4%, respectively. For the BSS estimators, their performance is worse than the RSS-I-RM (\( r = 1 \)) estimator, but are significantly better than the NMC estimator. In addition, the running time of all the estimators are comparable. These results consist with our analysis in Section 4 and Section 5.

From Table VI, we can see that RSS-II-BFS achieves the best relative variance on the FacebookLike dataset, followed by RSS-I-BFS, RSS-II-RM, RSS-I-RM, RSS-I-RM (\( r = 1 \)), the BSS estimators, and the NMC estimator. More specifically, the RSS-II-BFS estimator reduces the relative variance over the NMC estimator and the RSS-I-RM (\( r = 1 \)) estimators by 317% and 133%, respectively. The RSS-I-BFS estimator reduces the relative variance over NMC and RSS-I-RM (\( r = 1 \)) by 289% and 117%. Both RSS-I-RM and RSS-II-RM estimators cut the relative variance over NMC and RSS-I-RM (\( r = 1 \)) more than 231% and 184%, respectively. Similar to the result on the random graph dataset, all the BSS estimators are slightly worse than the RSS-I-RM (\( r = 1 \)) estimator but are significantly better than the NMC estimator. Also, the running time of all the estimators are comparable because the time complexities of all the estimators are \( O(Nm) \). These results confirm our analysis in the previous sections. Similar results can be observed in the Condmat (Table VII) and DBLP datasets (Table VIII).

To summarize, RSS-I-BFS and RSS-II-BFS achieve the best relative variance, and they reduce the relative variance over the existing estimators several times. The RSS estimators are better than the BSS estimators. The BSS/RSS estimators with the BFS edge-selection strategy are better than the BSS/RSS estimators with the random edge-

| Estimators  | Relative variance | Running time (s) |
|------------|------------------|-----------------|
| NMC        | 1.0000           | 0.3593          |
| RSS-I-RM (\( r = 1 \)) | 0.6723            | 0.3558          |
| RSS-I-RM   | 0.9429           | 0.3497          |
| RSS-I-BFS  | 0.8938           | 0.3748          |
| RSS-I-RM   | 0.3397           | 0.3373          |
| RSS-I-RM   | 0.2056           | 0.3783          |
| RSS-I-RM   | 0.9042           | 0.3633          |
| RSS-I-RM   | 0.9042           | 0.3749          |
| RSS-I-RM   | 0.3512           | 0.3716          |
| RSS-II-BFS | 0.2063           | 0.3847          |
Table VI. Results on FacebookLike dataset

| Estimators       | Relative variance | Running time (s) |
|------------------|-------------------|-----------------|
| NMC              | 1.0000            | 0.2007          |
| RSS-I-RM (r = 1) | 0.5585            | 0.2014          |
| BSS-I-RM         | 0.8898            | 0.2331          |
| BSS-I-BFS        | 0.6819            | 0.2354          |
| RSS-I-RM         | 0.3023            | 0.2002          |
| RSS-I-BFS        | 0.2570            | 0.2010          |
| BSS-II-RM        | 0.6947            | 0.2250          |
| BSS-II-BFS       | 0.6672            | 0.2284          |
| RSS-II-RM        | 0.2786            | 0.2027          |
| RSS-II-BFS       | 0.2397            | 0.2037          |

Table VII. Results on Condmat dataset

| Estimators       | Relative variance | Running time (s) |
|------------------|-------------------|-----------------|
| NMC              | 1.0000            | 1.2969          |
| RSS-I-RM (r = 1) | 0.7950            | 1.2958          |
| BSS-I-RM         | 0.9068            | 1.3043          |
| BSS-I-BFS        | 0.8531            | 1.3054          |
| RSS-I-RM         | 0.4883            | 1.2050          |
| RSS-I-BFS        | 0.1971            | 1.2411          |
| BSS-II-RM        | 0.8553            | 1.2513          |
| BSS-II-BFS       | 0.8421            | 1.3104          |
| RSS-II-RM        | 0.4891            | 1.2256          |
| RSS-II-BFS       | 0.2120            | 1.2284          |

Table VIII. Results on DBLP dataset

| Estimators       | Relative variance | Running time (s) |
|------------------|-------------------|-----------------|
| NMC              | 1.0000            | 8.5824          |
| RSS-I-RM (r = 1) | 0.5375            | 8.6536          |
| BSS-I-RM         | 0.9170            | 8.6292          |
| BSS-I-BFS        | 0.8373            | 8.8173          |
| RSS-I-RM         | 0.2100            | 8.3835          |
| RSS-I-BFS        | 0.1918            | 8.5933          |
| BSS-II-RM        | 0.9449            | 8.8825          |
| BSS-II-BFS       | 0.7997            | 9.1305          |
| RSS-II-RM        | 0.2003            | 8.8840          |
| RSS-II-BFS       | 0.1821            | 8.7052          |

selection strategy. All of our RSS estimators outperform the RSS-I-RM (r = 1) estimator. The proposed BSS estimators are slightly worse than the RSS-I-RM (r = 1) estimator, but still significantly outperform the NMC estimator. The running time of all the estimators are comparable.

**Scalability:** In order to study the scalability of various estimators, we generate synthetic probabilistic graphs $G$ with nodes ranging from 200,000 (200k) to 800,000 and the edges ranging from 800,000 to 3,200,000 (3.2m) according to the ER random graph model. And the probability of each edge is randomly generated according to a $[0, 1]$ uniform distribution. Also, for each estimator, we set the sample size $N$ to 1,000. Table IX shows the running time of different estimators on four large synthetic probabilistic graphs. As can be seen in Table IX, the running time increases as the size of the graph increases. In general, all the estimators achieve comparable running time, and they have linear growth w.r.t. the graph size. These results consist with the complexities of our estimators, i.e. $O(Nm)$.

**Effect of parameter $r$:** We study the effectiveness of the parameter $r$ in our proposed estimators on Condmat dataset. Similar results can be observed from other datasets. Fig. 3 and Fig. 4 show the relative variance of our type-I and type-II estimators w.r.t.
Table IX. Scalability: Running time on synthetic graphs. Here the two numbers in the 2nd-5th columns (eg. 200k/800k) indicate the numbers of nodes and edges respectively.

| Time (s) | 200k/800k | 400k/1.3m | 600k/1.6m | 800k/3.2m |
|----------|-----------|-----------|-----------|-----------|
| NMC      | 26.0820   | 156.9600  | 289.7720  | 365.0280  |
| NMC      | 153.9600  | 289.7720  | 365.0280  | 365.0280  |
| NMC      | 159.1990  | 286.2180  | 368.0910  | 368.0910  |
| NMC      | 25.2090   | 159.1990  | 286.2180  | 344.0180  |
| NMC      | 26.0820   | 156.9600  | 289.7720  | 365.0280  |
| BSS-I-RM | 25.2090   | 25.2090   | 25.2090   | 25.2090   |
| BSS-I-BFS | 27.2120  | 27.2120   | 27.2120   | 27.2120   |
| BSS-I-RM | 23.3430   | 23.3430   | 23.3430   | 23.3430   |
| BSS-I-BFS | 26.1450  | 26.1450   | 26.1450   | 26.1450   |
| BSS-II-RM | 29.5760  | 29.5760   | 29.5760   | 29.5760   |
| BSS-II-BFS | 25.2090  | 25.2090   | 25.2090   | 25.2090   |
| RSS-I-RM | 25.2090   | 25.2090   | 25.2090   | 25.2090   |
| RSS-I-BFS | 27.2120  | 27.2120   | 27.2120   | 27.2120   |
| RSS-II-RM | 26.4440  | 26.4440   | 26.4440   | 26.4440   |
| RSS-II-BFS | 26.4990  | 26.4990   | 26.4990   | 26.4990   |

Fig. 3. Effect of $r$ of BSS-I/RSS-I estimators.

Various $r$. As can be seen in Fig. 3, the BSS-I estimators exhibit similar relative variance over different $r$ values. However, the relative variance of the RSS-I-RM estimator decreases as the $r$ increases when $r \leq 5$, and otherwise it increases as the $r$ increases. For the RSS-I-BFS estimator, the relative variance decreases as $r$ increases, and when $r \geq 5$ the descent rate is very small, and the curve tends to be smooth. Based on this observation, $r = 5$ is the best choice, which is used in the previous experiments. For the type-II estimators, we test the parameter $r$ from 10 to 70, and the results (Fig. 4) show that all of our type-II estimators except RSS-II-RM are not very sensitive w.r.t. the parameter $r$. An exception, the relative variance of the RSS-I-BFS estimator decreases as the $r$ increases when $r \leq 50$, and when $r \geq 50$ the curve tends to be smooth. Therefore, $r = 50$ is a good choice. In our previous experiments, we set $r$ to 50. Table X and Table XI report the running time of type-I estimators and type-II estimators under different $r$ values. We can see that the running time of both type-I estimators and type-II estimators are comparable.

Effect of sample size: As shown in the previous experiments, the RSS-I-BFS and the RSS-II-BFS estimators are the best two estimators. Here we study how sample size affects the estimating accuracy of these two estimators on the Condmat dataset. Similar results can be observed on the other dataset. Fig. 5 shows the relative variance of the estimators under various sample size. As can be observed in Fig. 5, the curves
of RSS-I-BFS and RSS-II-BFS estimators are very smooth, which indicate that the relative variance of both RSS-I-BFS and RSS-II-BFS estimators are robust w.r.t. the sample size.

7. RELATED WORK

After the seminal work by Kempe, et al. [Kempe et al. 2003], influence maximization in social networks has recently attracted much attention in data mining and social network analysis research communities [Kempe et al. 2005; Leskovec et al. 2007; Chen et al. 2009; Chen et al. 2010; Goyal et al. 2010; Chen et al. 2011; Goyal et al. 2011]. A crucial subroutine in influence maximization is the influence function evaluation to which the influenceability estimation problem presented in this paper is closely related. In the following, we first review some notable work on influence maximization problem, and then discuss the existing work on influence function evaluation. In [Leskovec et al. 2007], the authors study the influence maximization problem under the context of water distribution and blogosphere monitoring. They propose a so-called CELF framework for optimizing the influence maximization algorithms. To further accelerate the influence maximization algorithms, Chen, et al. in [Chen et al. 2009] propose a scalable algorithm by sampling $N$ possible graphs and estimating the influence spread of all vertices on each possible
graph at one time. Subsequently, the same authors propose a series of scalable algorithms in [Chen et al. 2010] and [Chen et al. 2011] for influence maximization by developing the heuristic vertices-selection strategies on unsigned and signed networks, respectively. Recently, Goyal, et al. in [Goyal et al. 2010] [Goyal et al. 2011] consider the problem of learning the influence probabilities, and study the influence maximization from a data-driven perspective. Note that all the mentioned methods focus on the influence maximization problem. For the influence function evaluation problem, Kempe, et al. firstly pose it as an open problem in [Kempe et al. 2005]. Then, Chen, et al. in [Chen et al. 2010] show that the influence function evaluation problem is #P complete. Given the hardness of the problem, most of the existing work for this problem, such as [Kempe et al. 2003] [Leskovec et al. 2007] [Chen et al. 2009]; [Chen et al. 2010], are based on the Naive Monte-Carlo (NMC) sampling. In this paper, we study the influenceabilty evaluation problem and develop more accurate RSS estimators for estimating the influenceability, and our algorithms can also be used for influence function evaluation.

Our work is also related to the uncertain graph mining. Recently, uncertain graphs mining have been attracted increased interest because of the increasing applications in biological database [Sevon et al. 2006], network routing [Ghosh et al. 2007], and influence networks [Goyal et al. 2011]. There are a large body of works have been proposed in the literature. Notable work includes finding the reliable subgraph in a large uncertain graph [Hintsanen and Toivonen 2008]; [Jin et al. 2011a], frequent subgraph mining in uncertain graph database [Zou et al. 2010a]; [Zou et al. 2010b], subgraph search in large uncertain graph [Yuan et al. 2011], K-nearest neighbor search in uncertain graph [Potamias et al. 2010], and distance constraint reachability computation in uncertain graph [Jin et al. 2011b]. In general, all the mentioned uncertain graph mining problems are shown to be #P-complete, and thereby finding the exact solution is intractable in large uncertain graphs. Consequently, most existing work, such as [Potamias et al. 2010] and [Jin et al. 2011a], are based on NMC sampling. Basically, the NMC sampling based methods lead to a large variance, thus reduce the performance of the algorithms. Recently, Jin, et al. in [Jin et al. 2011b] propose a recursive stratified sampling method for distance-constraint reachability computation on uncer-
tain graph, although they do not claim their method is a stratified sampling. It is important to note that their method is a very special case of our RSS-I algorithm. In their method, they select only one edge for stratification at a time, and then recursively perform this procedure. Unlike their algorithm, first, we develop a generalized algorithm (RSS-I) that selects $r$ edges for stratification. Second, unlike their reachability problem, here we study the influenceability evaluation problem using the RSS-I sampling. Moreover, in our work, we also develop another RSS estimator, i.e. RSS-II estimator. Note that all of our RSS estimators can also be applied into the distance-constraint reachability computation problem.

In addition, our work is related to the network reliability estimation problem, where a network is modeled as an uncertain graph and the goal is to estimate some reliability metrics of the network [Fishman 1986a; Rubino 1999]. There are many work on this topic in the last five decades. Surveys can be found in [Colbourn 1987; Rubino 1999].

Below, we review the Monte-Carlo algorithms for network reliability estimation. Kumamoto, et al. [Kumamoto et al. 1977] propose an efficient Monte-Carlo algorithm by exploiting the bound of the reliability metric. Fishman [Fishman 1986b] proposes a more generalized Monte-Carlo algorithm based on such bound techniques for reliability estimation. Subsequently, Fishman [Fishman 1986a] compares four Monte-Carlo algorithms for network reliability estimation problem. Cancela, et al. in [Cancela and Khadiri 2003] propose a recursive variance-reduction algorithm for network reliability estimation. Note that all the mentioned Monte-Carlo algorithms are tailored for the network reliability estimation problem, and the reliability measure is typically a Boolean metric thus they cannot be used in our problem.

8. CONCLUSIONS

In this paper, we focus on the influenceability evaluation problem, which is a fundamental issue for influence analysis in social network. This problem is known to be $\#P$-complete, and the only existing algorithm is based on the Naive Monte-Carlo (NMC) sampling. To reduce the variance of the NMC estimator, we propose two basic stratified sampling (BSS) estimators. Furthermore, based on our BSS estimators, we present two recursive stratified sampling (RSS) estimators. We conduct comprehensive experiments on one synthetic and three real datasets, and the results confirm that our RSS estimators reduce the variance of the NMC estimator by several times. There are several future directions that deserve further investigation. First, most of our estimators except the RSS estimators with BFS edge selection do not take the graph structural information into account. In our experiments, the RSS estimators with BFS edge selection are shown much better performance than the RSS estimators with random edge selection. A promising direction is to exploit the graph structural information to develop more efficient and more accurate estimators for influenceability evaluation. Second, our estimation techniques are quite general. For many uncertain graph mining problems, such as shortest path [Potamias et al. 2010], reachability [Jin et al. 2011b], and reliable subgraph discovery [Jin et al. 2011a], our estimators can be directly used. For these problems, we only need to replace the $\phi_s(G_F)$ to other quantities, such as the length of the shortest path, the reachability function between two nodes, and the reliable subgraph metric. Most of these uncertain graph mining problems are based on NMC. Another promising future direction is to apply our estimation techniques to these problems.

REFERENCES

Cancela, H. and Khadiri, M. E. 2003. The recursive variance-reduction simulation algorithm for network reliability evaluation. IEEE Transactions on Reliability 52, 2, 207–212.
CHEN, W., COLLINS, A., CUMMINGS, R., KE, T., LIU, Z., RINCÓN, D., SUN, X., WANG, Y., WEI, W., AND YUAN, Y. 2011. Influence maximization in social networks when negative opinions may emerge and propagate. In SDM.

CHEN, W., WANG, C., AND WANG, Y. 2010. Scalable influence maximization for prevalent viral marketing in large-scale social networks. In KDD.

CHEN, W., WANG, Y., AND YANG, S. 2009. Efficient influence maximization in social networks. In KDD.

COLBOURN, C. J. 1987. The Combinatorics of Network Reliability. Oxford University Press, Inc., New York, NY, USA.

FISHMAN, G. S. 1986a. A comparison of four monte carlo methods for estimating the probability of s-t connectedness. IEEE Transactions on Reliability 35, 2, 145–155.

FISHMAN, G. S. 1986b. A monte carlo sampling plan for estimating network reliability. Operations Research 34, 4, 581–594.

GHOSH, J., NGO, H. Q., YOON, S., AND QIAO, C. 2007. On a routing problem within probabilistic graphs and its application to intermittently connected networks. In INFOCOM.

GOYAL, A., BONCHI, F., AND LAKSHMANAN, L. V. S. 2010. Learning influence probabilities in social networks. In WSDM.

GOYAL, A., BONCHI, F., AND LAKSHMANAN, L. V. S. 2011. A data-based approach to social influence maximization. PVLDB 5, 1, 73–84.

HINTSANEN, P. AND TOIVONEN, H. 2008. Finding reliable subgraphs from large probabilistic graphs. Data Min. Knowl. Discov. 17, 1, 3–23.

JIN, R., LIU, L., AND AGGARWAL, C. C. 2011a. Discovering highly reliable subgraphs in uncertain graphs. In KDD.

JIN, R., LIU, L., DING, B., AND WANG, H. 2011b. Distance-constraint reachability computation in uncertain graphs. PVLDB 4, 9, 551–562.

KEMPE, D., KLEINBERG, J. M., AND TARDOS, É. 2003. Maximizing the spread of influence through a social network. In KDD.

KEMPE, D., KLEINBERG, J. M., AND TARDOS, É. 2005. Influential nodes in a diffusion model for social networks. In ICALP.

KUMAMOTO, H., TANAKA, K., AND INOUE, K. 1977. Efficient evaluation of system reliability by monte carlo method. IEEE Transactions on Reliability 26, 5, 311–315.

LAPPAS, T., TERZI, E., GUNOPULOS, D., AND MANNILA, H. 2010. Finding effectors in social networks. In KDD.

LESKOVEC, J., KRAUSE, A., GUESTRIN, C., FALOUTSOS, C., VANBRIESEN, J. M., AND GLANCE, N. S. 2007. Cost-effective outbreak detection in networks. In KDD.

LIU, H., SALERNO, J., AND YOUNG, M. J. 2009. Social Computing, Behavioral Modeling, and Prediction. Springer; 1 edition.

NEWMAN, M. 2001. The structure of scientific collaboration networks. Proc. Natl. Acad. Sci. USA 98, 404–409.

OPSahl, T. AND PANZARASA, P. 2009. Clustering in weighted networks. Social Networks 31, 2, 155–163.

POTAMIAS, M., BONCHI, F., GIONIS, A., AND KOLLIOS, G. 2010. k-nearest neighbors in uncertain graphs. PVLDB 3, 1, 997–1008.

RUBINO, G. 1999. Network reliability evaluation. 275–302.

SEVON, P., ERONEN, L., HINTSANEN, P., KULOVESI, K., AND TOIVONEN, H. 2006. Link discovery in graphs derived from biological databases. In DILS.

TANG, J., SUN, J., WANG, C., AND YANG, Z. 2009. Social influence analysis in large-scale networks. In KDD.

THOMPSON, S. K. 2002. Sampling. Wiley-Interscience; 2 edition.

YUAN, Y., WANG, G., WANG, H., AND CHEN, L. 2011. Efficient subgraph search over large uncertain graphs. PVLDB 4, 11, 876–886.

ZHOU, Y., CHENG, H., AND YU, J. X. 2010. Clustering large attributed graphs: An efficient incremental approach. In ICDM. 689–698.

ZOU, Z., GAO, H., AND LI, J. 2010a. Discovering frequent subgraphs over uncertain graph databases under probabilistic semantics. In KDD. 633–642.

ZOU, Z., LI, J., GAO, H., AND ZHANG, S. 2010b. Mining frequent subgraph patterns from uncertain graph data. IEEE Trans. Knowl. Data Eng. 22, 9, 1203–1218.