On Random Walk Based Graph Sampling

Rong-Hua Li[†], Jeffrey Xu Yu[*], Lu Qin[‡], Rui Mao[†], and Tan Jin[#]

[†]*Shenzhen University, Shenzhen, China*
[*]*The Chinese University of Hong Kong, Hong Kong*
[‡]*Centre for QCIS, FEIT, University of Technology, Sydney, Australia*
[#]*Northeastern University, Shenyang, China*
{rhli,yu}@se.cuhk.edu.hk, Lu.Qin@uts.edu.au, mao@szu.edu.cn, alextanjin@gmail.com

***Abstract*—Random walk based graph sampling has been recognized as a fundamental technique to collect uniform node samples from a large graph. In this paper, we first present a comprehensive analysis of the drawbacks of three widely-used random walk based graph sampling algorithms, called re-weighted random walk (RW) algorithm, Metropolis-Hastings random walk (MH) algorithm and maximum-degree random walk (MD) algorithm. Then, to address the limitations of these algorithms, we propose two general random walk based algorithms, named rejection-controlled Metropolis-Hastings (RCMH) algorithm and generalized maximum-degree random walk (GMD) algorithm. We show that RCMH balances the tradeoff between the limitations of RW and MH, and GMD balances the tradeoff between the drawbacks of RW and MD. To further improve the performance of our algorithms, we integrate the so-called delayed acceptance technique and the non-backtracking random walk technique into RCMH and GMD respectively. We conduct extensive experiments over four real-world datasets, and the results demonstrate the effectiveness of the proposed algorithms.**

I. Introduction

Online social network analysis has attracted extensive attention in recent years. A fundamental problem in online social network analysis is estimating the nodal or topological characteristics of the network [1]. However, in the context of online social networks, this problem is very challenging [2]. The reason is because in most online social networking platforms, such as Facebook (http://www.facebook.com/) and Twitter (http://twitter.com/), the network topology is typically unknown to the researchers. Furthermore, in most scenarios, the size of the network is also unknown [1], [2]. As a result, it is impossible to perform "independent random sampling" over the nodes of the network. To overcome this problem, many graph sampling via crawling techniques have been widely-used [3], [4]. These approaches can be roughly classified into two categories: graph-traversal based methods [3], [5], [6], [4] and random walk based methods [7], [8], [9], [10], [1]. Graph-traversal based methods apply the breadth-first search (BFS) or the depth-first search (DFS) algorithm to collect nodes. These algorithms, however, typically introduce a bias towards high-degree nodes [4]. Moreover, such a bias is unknown, and it is also very hard to analyze in general graphs [4]. Instead, the random walk based methods are very popular for graph sampling, because they can produce unbiased samples or generate samples with a known bias.

In the literature, there are three widely-used random walk based sampling algorithms: the re-weighted random walk (RW) [9], [1], [8], the Metropolis-Hastings random walk (MH) [10], [1], and the maximum-degree random walk (MD) [7], [11]. Specifically, RW first performs a traditional random walk on a graph to collect nodes, and then constructs an unbiased estimator by using a re-weighting strategy [1]. As shown in Section II, the effectiveness of RW relies on the similarity between the stationary distribution of the random walk (proportional to the node degree) and the uniform distribution. However, in many real-world graphs, the stationary distribution of the random walk is very far from the uniform distribution. Thus, RW suffers from the *large deviation* problem, i.e., the deviation between the stationary distribution of the random walk and the uniform distribution is very large. MH [10], [1] performs a Metropolis-Hastings random walk on a graph to collect node samples which produces a uniform stationary distribution. This algorithm, however, must reject a large number of nodes to get the uniform samples. The performance of this algorithm is therefore dependent on its sample acceptance ratio. However, as observed in our experiments, the acceptance ratio of MH is typically very low in real-world networks. Therefore, MH generally suffers from the sample rejection problem. MD runs a maximum-degree random walk to gather nodes [7]. Unlike the traditional random walk, in each step, the maximum-degree random walk selects a neighbor node with probability $1/d_{\max}$ where $d_{\max}$ is the maximum degree of the nodes in a graph. For any node u with degree d_u, this process is equal to the process of adding $d_{\max} - d_u$ self-loops on u and then performing the traditional random walk on such a modified graph. As shown in [7], the stationary distribution of the maximum-degree random walk is the uniform distribution. However, to achieve the uniform stationary distribution, MD needs to traverse the self-loops, thus resulting in a large number of repeated samples. Too many repeated samples generally leads to a large variance for the random walk based algorithms [2]. Consequently, MD suffers from the repeated samples problem.

Our contributions. To address the limitations in previous algorithms, in this work, we propose two novel random walk based graph sampling algorithms. To the best of our knowledge, we are the first group to systematically analyze the drawbacks of the existing random walk based graph sampling algorithms, and present new solutions to balance the tradeoffs of these drawbacks. More specifically, to balance the tradeoff between the *large deviation* problem of RW and sample rejection problem of MH, we propose a rejection-controlled Metropolis-Hastings (RCMH) algorithm. We show, in both theory and experiments, that RCMH can mitigate the *large deviation* problem of RW and reduce the sample-rejection ratio of MH simultaneously by setting an appropriate parameter, thus providing a good tradeoff between the limitations of RW and MH. Moreover, we show that both RW and MH are two special cases of RCMH, thus our algorithm establishes a connection

∗ Dr. Rui Mao is the corresponding author.

between these two widely-used graph sampling algorithms. On the other hand, we devise a generalized maximum-degree random walk (GMD) algorithm to balance the tradeoff between the *large deviation* problem of RW and the repeated samples problem of MD. We prove that by setting an appropriate parameter, GMD can not only alleviate the *large deviation* problem of RW, but it is also expected to produce a smaller number of repeated samples compared to MD. GMD is also very general which treats RW and MD as two special cases, thus creating an interesting link between RW and MD.

We present several optimization techniques to further improve the estimation accuracy of RCMH and GMD. Specifically, we integrate the so-called *delayed acceptance* technique developed in [2] into RCMH to reduce the asymptotic variance [12]. For GMD, we propose a non-backtracking GMD algorithm, called NGMD, which integrates the non-backtracking random walk technique [2]. Based on the non-backtracking random walk, we show that NGMD not only preserves the same stationary distribution as GMD, but it also reduces the asymptotic variance [12] of GMD. Finally, we conduct extensive experiments over four real-world graphs to evaluate the proposed algorithms. The results show that our algorithms significantly outperform the state-of-the-art algorithm for unbiased graph sampling by setting appropriate parameters. We also provide an empirical recommendation on setting the parameters of our algorithms to achieve good tradeoffs of the limitations of the previous algorithms.

Organization. The rest of this paper is organized as follows. Below, we will further review the previous research related to ours. After that, we present a detailed analysis of the limitations of RW, MH and MD in Section II. The RCMH and GMD algorithms are presented in Section III and Section IV, respectively. In Section V, we propose several optimization techniques to improve the estimation accuracy of RCMH and GMD. Extensive experimental studies are reported in Section VI. We conclude this work in Section VII.

Further related work. Graph sampling via crawling has gained much interest in recent years. Except the methods discussed previously, a recent notable work is [13] where the authors propose an elegant random walk based algorithm, aiming at reducing the mixing time of the random walk. This method is very useful when one wants to generate independent samples (i.e., performing a random walk to generate one sample). However, in many real-world applications, such as degree distribution estimation [1], [2], clustering coefficient estimation [14], size estimation [15], and average degree estimation [16], it has turned out that using dependent samples (i.e., all samples are generated by only one random walk) is good enough to construct a very accurate estimator. On the other hand, many online social networks are known to have low mixing time [17], [14], [18]. Therefore, reducing the mixing time may not significantly improve the effectiveness of the random walk based algorithms for such applications. In this work, we propose two novel algorithms to balance the tradeoffs of the limitations of the existing random walk based algorithms. Our work is complementary to [13], and the techniques in [13] could also be integrated in our algorithms.

Besides graph sampling via crawling, there are some other graph sampling algorithms [3], [6], [19]. For example, Leskovec and Faloutsos [3] compared a lot of graph sampling algorithms based on a known graph topology. In [19], Maiya and Berger-Wolf proposed an expansion-based sampling algorithm to sample a subgraph so that it can preserve the community structures. Subsequently, the same authors presented a detailed study on the benefits of biases in different sampling algorithms [6]. However, most of these sampling algorithms can only work on the known graph topology. Our algorithms, however, can work on an unknown graph topology, which is clearly more practical for sampling online social networks. Another line of research is to devise random walk based graph sampling algorithms to handle the disconnected graphs. Such algorithms include the parallel random walks method [1], the multidimensional random walk method [20], and the random walk with jump algorithm [21]. The proposed algorithms are complementary to these approaches, and these techniques can also be combined into our algorithms.

II. Background and Motivation

Consider an undirected graph $G = (V, E)$ with $n = |V|$ nodes and $m = |E|$ edges. Denote by $N(u)$ the set of neighbors of node $u \in V$, and by $d_u = |N(u)|$ the degree of u. Following the standard notations in graph sampling literature [9], [2], the unbiased graph sampling problem is to estimate the following quantity:

$$\mathbb{E}_{\pi^{\mathbf{u}}}(f) \triangleq \sum\nolimits_{u \in V} f(u)/n, \tag{1}$$

where $\pi^{\mathbf{u}} = [1/n, \cdots, 1/n]$ denotes a uniform distribution and $f : V \to \mathbb{R}$ is a real-valued function defined on the node set V. Notice that the function f can characterize different nodal or topological properties of a graph depending on different definitions [9], [2]. For example, if f is defined by $f(u) = d_u$ for all $u \in V$, then the quantity $\mathbb{E}_{\pi^{\mathbf{u}}}(f)$ denotes the average degree of the graph G. If f is defined by $f(u) = \mathbf{1}_{\{d_u = d\}}$ for all $u \in V$ given that $d = 1, \cdots, n-1$, then $\mathbb{E}_{\pi^{\mathbf{u}}}(f)$ denotes the degree distribution of G. Here $\mathbf{1}_{\{d_u = d\}}$ is an indicator function, i.e., if $d_u = d$, $\mathbf{1}_{\{d_u = d\}} = 1$, $\mathbf{1}_{\{d_u = d\}} = 0$ otherwise.

In this paper, we focus on random walk based algorithms for estimating $\mathbb{E}_{\pi^{\mathbf{u}}}(f)$. In the literature, there are three random walk based algorithms that can produce unbiased estimators for $\mathbb{E}_{\pi^{\mathbf{u}}}(f)$. These algorithms are the Re-weighted random walk algorithm (RW) [9], [8], [1], the Metropolis-Hastings random walk algorithm (MH) [10], [1], and the maximum-degree random walk algorithm (MD) [7], [11]. Below, we review these algorithms and discuss their pros and cons as well.

Re-weighted random walk (RW). The RW algorithm performs a random walk on graph to collect nodes [9], [1]. It is well-known that, in a connected, aperiodic and undirected graph, the stationary distribution of a random walk is proportional to the node degree [22], i.e., $\pi^{\mathbf{rw}}(u) = d_u/2m$ for all $u \in V$. Based on this fact, the nodes collected by the random walk are biased toward high-degree nodes. To correct such bias, the RW algorithm makes use of a re-weighting strategy which can be done using the well-known Hanse-Hurwitz estimator [8], [9], [23], [1]. In particular, the estimator in RW is given by

$$\mathbb{E}_{\pi^{\mathbf{rw}}}(f) = \frac{\sum_{u \in S} f(u) w^{\mathbf{rw}}(u)}{\sum_{u \in S} w^{\mathbf{rw}}(u)}, \tag{2}$$

where S is the set of sampled nodes and $w^{\mathbf{rw}}(u) \propto 1/d_u$ denotes the weight of node u. Note that here the weight $w^{\mathbf{rw}}(u)$ can be up to any multiplicative constant.

It was shown that the above estimator can be interpreted using the importance sampling (IS) framework [9], [23]. Specifically, instead of drawing nodes from the target distribution, the IS framework samples nodes from a different and easily implemented trial distribution [12]. In the RW algorithm, the target distribution is the uniform distribution $\pi^{\mathbf{u}}$, and the trial distribution is $\pi^{\mathbf{rw}}$. According to the IS framework [12], the importance weight for a node u is given by $w^{\mathbf{rw}}(u) \triangleq \pi^{\mathbf{u}}(u)/\pi^{\mathbf{rw}}(u) = 2m/nd_u \propto 1/d_u$ meeting the definition in Eq. (2). It was turned out that the estimator $\mathbb{E}_{\pi^{\mathbf{rw}}}(f)$ is asymptotically unbiased [12], i.e., $\mathbb{E}_{\pi^{\mathbf{rw}}}(f) \to \mathbb{E}_{\pi^{\mathbf{u}}}(f)$ as $n \to \infty$. The variance of the estimator $\mathbb{E}_{\pi^{\mathbf{rw}}}(f)$ depends on the variance of $f(u)w^{\mathbf{rw}}(u)$. If $f(u)$ is uncorrelated with $w^{\mathbf{rw}}(u) = \pi^{\mathbf{u}}(u)/\pi^{\mathbf{rw}}(u)$, such variance relies on the similarity between $\pi^{\mathbf{u}}(u)$ and $\pi^{\mathbf{rw}}(u)$. Intuitively, the closer the trial distribution $\pi^{\mathbf{rw}}$ and the target distribution $\pi^{\mathbf{u}}$ are, the lower the variance of the estimator $\mathbb{E}_{\pi^{\mathbf{rw}}}(f)$. Liu in [12] proposes a "rule of thumb" to quantify this intuition which can also be used to measure the effectiveness of the IS framework. Specifically, assume that there is an estimator $\hat{\mathbb{E}}$ using N independent samples from the target distribution q ($q = \pi^{\mathbf{u}}$ in our case). Then, Liu's "rule of thumb" states that the number of samples from the trial distribution p ($p = \pi^{\mathbf{rw}}$ in our case) is at most $N(1 + \mathrm{var}_p(q(X)/p(X)))$ to achieve the same variance as that of $\hat{\mathbb{E}}$ [12], [11], [24], [25]. As a consequence, we strive to seek a trial distribution p such that $\mathrm{var}_p(q(X)/p(X))$ (also called the χ-distance between q and p [12]) is as small as possible.

In RW, the trial distribution $\pi^{\mathbf{rw}}$ is proportional to the degree of nodes, while the target distribution is the uniform distribution $\pi^{\mathbf{u}}$. As shown in our experiments, $\pi^{\mathbf{rw}}$ is typically far from the uniform distribution $\pi^{\mathbf{u}}$ in many real-world graphs. That is to say, there is a *large deviation* between the trial and target distributions of RW in many real-world graphs. By Liu's "rule of thumb", the effectiveness of RW depends on the similarity between $\pi^{\mathbf{u}}$ and $\pi^{\mathbf{rw}}$. Therefore, in many real-world graphs, RW suffers from the *large deviation* problem.

Metropolis-Hastings random walk (MH). The MH algorithm is an application of the Metropolis-Hastings algorithm [26], [27] for unbiased graph sampling. It makes use of a modified random walk to draw nodes from the graph. In particular, it modifies the transition probabilities of a random walk so that the walk converges into a uniform distribution. The transition probability of MH is given by

$$P_{uv}^{\mathbf{mh}} = \begin{cases} 1/d_u \times \min\{1, d_u/d_v\}, & if\ v \in N(u), \\ 1 - \sum_{u \neq w} P_{uw}^{\mathbf{mh}}, & if\ u = v, \\ 0, & otherwise, \end{cases} \tag{3}$$

where $\min\{1, d_u/d_v\}$ is the acceptance function of MH. Let $v \in N(u)$ be a neighbor of node u. When MH draws a node v from $N(u)$ with probability $1/d_u$, MH accepts v as a sample with probability $\min\{1, d_u/d_v\}$, and reject it with probability $1 - \min\{1, d_u/d_v\}$. It has turned out that the stationary distribution of MH is $\pi^{\mathbf{mh}} = \pi^{\mathbf{u}} = [1/n, \cdots, 1/n]$ [10], [1]. Therefore, the sample mean obtained by MH can be directly used to construct an unbiased estimator for $\mathbb{E}_{\pi^{\mathbf{u}}}(f)$.

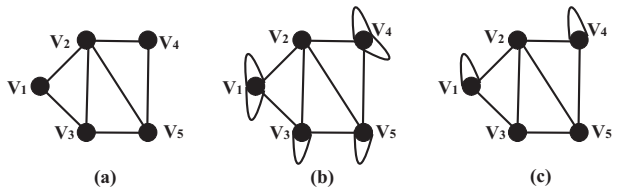

Fig. 1: Running example

Using the language in IS framework, the *trial distribution* of MH is equivalent to the target distribution, i.e., $\pi^{\mathbf{u}} = \pi^{\mathbf{mh}}$. This suggests that MH can overcome the *large deviation* problem of RW. However, this is not free. To obtain the uniform samples, MH has to reject a considerable number of samples. In other words, to achieve the same sample size as that of RW, MH generally needs to draw a lot more nodes than RW. In the setting of unbiased graph sampling, the primary costs of different algorithms are measured by the number of nodes drawn by various algorithms [1]. If we fix the number of nodes that different algorithms need to draw, then the number of final samples obtained by MH is much smaller than that of RW, which clearly degrades the performance of MH. Hence, the MH algorithm suffers from the sample rejection problem.

Maximum-degree random walk (MD). The MD algorithm runs a random walk on a dynamically created regular graph to collect nodes [7], [11]. The idea is that the algorithm modifies the original graph into a regular graph by adding self-loops on the nodes so that the degree of each node equals the maximum degree of the original graph. Note that this can be done implicitly and on-the-fly. Specifically, when the walk traverses a node u, the walk selects a neighbor node from $N(u)$ with probability $1/d_{\max}$, where $d_{\max}$ denotes the maximum degree of the original graph. Following this process, in each step, the walk stays at u with probability $(d_{\max} - d_u)/d_{\max}$. That is to say, the walk traverses the self-loop with probability $(d_{\max} - d_u)/d_{\max}$. Fig. 1(a) and Fig. 1(b) illustrate the original graph and the corresponding regular graph respectively. For the graph in Fig. 1(a), the maximum degree is 4. Hence, each node in the corresponding regular graph shown in Fig. 1(b) has a degree of 4 including self-loops. The maximum-degree random walk on a graph is equivalent to the traditional random walk on the corresponding regular graph. This fact indicates that the stationary distribution of the maximum-degree random walk is a uniform distribution. As a result, the sample mean obtained by MD can be directly utilized to construct an unbiased estimator for $\mathbb{E}_{\pi^{\mathbf{u}}}(f)$ [7]. Similarly, using the language in IS framework, the *trial distribution* of MD is equal to the target distribution $\pi^{\mathbf{u}}$. Therefore, MD can circumvent the *large deviation* problem of RW.

MD also has its drawbacks. First, the algorithm will produce many repeated samples, because MD needs to traverse the self-loops. This issue worsens when the walk traverses the low-degree nodes as these nodes must add many self-loops. Note that too many repeated samples typically result in a large variance for the random walk based samplers [2]. Thus, MD suffers from the repeated samples problem. Second, MD requires the knowledge of maximum degree which is impractical in the context of unbiased graph sampling via crawling. To overcome this problem, Bar-Yossef et al. [7] propose to set a very large constant as the maximum degree. Clearly, this

process aggravates the repeated samples problem if such a constant is larger than the maximum degree.

The above analysis motivates us to ask the following questions. Can we have a sampling algorithm that achieves a tradeoff between the *large deviation* problem of RW and the sample rejection problem of MH? And can we devise an algorithm that can balance the *large deviation* problem of RW and the repeated samples problem of MD? In the following two sections, we shall present two novel algorithms to achieve these purposes.

III. The RCMH algorithm

In this section, we propose a novel rejection-controlled Metropolis-Hastings (RCMH) algorithm which balances the tradeoff between the *large deviation* problem of RW and the sample rejection problem of MH. Below, we first describe the RCMH algorithm in a general form and analyze its theoretical properties. Then, we show how to apply the RCMH algorithm to unbiased graph sampling.

A. Rejection-controlled Metropolis-Hastings Algorithm

The key idea of the rejection-controlled Metropolis-Hastings (RCMH) algorithm is as follows. First, we devise a novel Metropolis-Hastings-type algorithm called RCMH algorithm whose acceptance function is parameterized by a parameter $\alpha \in [0,1]$. Such a newly-devised acceptance function allows us to improve the acceptance ratio of the original Metropolis-Hastings algorithm. Note that, after modifying the acceptance function, the stationary distribution of the RCMH algorithm is no longer the target distribution of the original Metropolis-Hastings algorithm. Second, we prove that the χ-distance between the stationary distribution of the RCMH algorithm and the target distribution of the original Metropolis-Hastings algorithm is smaller than the χ-distance between the trial and target distribution of the original Metropolis-Hastings algorithm. The detailed description of RCMH is given below.

First, let us define some useful notations. Let $\mathcal{U}$ be a finite space, p be the probability distribution on $\mathcal{U}$. The support of p is defined as $supp(p) = \{u \in \mathcal{U} | p(u) > 0\}$. Let p and π be the trial and target distributions of the original Metropolis-Hastings algorithm, respectively. Denote by P the initial Markov Chain of the original Metropolis-Hastings algorithm with stationary distribution p. By the basic assumption of the Metropolis-Hastings algorithm [27], [12], [11], P is ergodic and it satisfies the following conditions: (1) $P(u,v) > 0 \Leftrightarrow P(v,u) > 0$ for all $u, v \in \mathcal{U}$, and (2) $supp(p) = supp(\pi)$. Further, we assume that P is time-reversible, i.e., $p(u)P(u,v) = p(v)P(v,u)$ (this is indeed the case in unbiased graph sampling). The RCMH algorithm is identical to the original Metropolis-Hastings algorithm, except its acceptance function $r(u,v,\alpha)$ which is defined by

$$r(u,v,\alpha) = \min\{(\frac{\pi(v)P(v,u)}{\pi(u)P(u,v)})^{\alpha}, 1\} = \min\{(\frac{\pi(v)p(u)}{\pi(u)p(v)})^{\alpha}, 1\}, \tag{4}$$

where $\alpha \in [0,1]$ is the rejection-controlled parameter, π is the target distribution of the original Metropolis-Hastings algorithm, and the second equality is due to the time-reversible property of P. Clearly, the acceptance function $r(u,v,\alpha)$ is a monotonically decreasing function w.r.t. the parameter α. If $\alpha \in [0,1)$, we have $r(u,v,\alpha) > r(u,v,1)$. It is important to note that when $\alpha = 1$, the acceptance function is the same as that of the original Metropolis-Hastings algorithm. Therefore, in this case, the RCMH algorithm becomes the original Metropolis-Hastings algorithm. Obviously, at any state $u \in \mathcal{U}$, the acceptance ratio of the RCMH algorithm ($r(u,v,\alpha)$) is larger than that of the original Metropolis-Hastings algorithm ($r(u,v,1)$) when $\alpha \in [0,1)$. In addition, when $\alpha = 0$, the acceptance function always equals 1 (in this case, the next state is always accepted); therefore, the Markov Chain of the RCMH algorithm is the same as the initial Markov Chain P.

Based on the acceptance function $r(u,v,\alpha)$, the probability transition matrix of the Markov Chain of the RCMH algorithm is given by

$$P_{\mathbf{rcmh}}(u,v) = \begin{cases} P(u,v)r(u,v,\alpha), & if\ u \neq v, \\ 1 - \sum_{u \neq w} P_{\mathbf{rcmh}}(u,w), & otherwise. \end{cases} \tag{5}$$

The following theorem shows that the Markov Chain of the RCMH algorithm is ergodic, and thus it has a unique stationary distribution denoted by $\pi^{\mathbf{rcmh}}$.

Theorem 3.1: $P_{\mathbf{rcmh}}$ forms an ergodic Markov Chain and its unique stationary distribution $\pi^{\mathbf{rcmh}}$ meets the following condition: for any pair of states $u, v \in supp(\pi^{\mathbf{rcmh}})$, $\pi^{\mathbf{rcmh}}(u)/\pi^{\mathbf{rcmh}}(v) = p(v,u)(\pi(u)p(u,v))^{\alpha}/(p(u,v)(\pi(v)p(v,u))^{\alpha})$ holds. Further, assume that $p(u,v) = p(u,w)$ holds, for any $u, v, w \in supp(\pi^{\mathbf{rcmh}})$. Then, $\pi^{\mathbf{rcmh}}$ is given by $\pi^{\mathbf{rcmh}}(u) = (\pi(u)p(u,v))^{\alpha}/(Z \times p(u,v))$, where Z is a normalization constant.

Proof: The proof is omitted due to space limit. ■

Next, we prove that the χ-distance [12] between the target distribution π and $\pi^{\mathbf{rcmh}}$ is smaller than the χ-distance between π and p. Specifically, we have the following theorem.

Theorem 3.2: $\mathrm{var}_{\pi^{\mathbf{rcmh}}}(\frac{\pi(u)}{\pi^{\mathbf{rcmh}}(u)}) \leq \mathrm{var}_{p}(\frac{\pi(u)}{p(u)})$.

Proof: By our assumptions, one can easily deduce that $supp(\pi^{\mathbf{rcmh}}) = supp(\pi) = supp(p)$. Based on this, we have

$$\begin{aligned} &\mathbb{E}_{\pi^{\mathbf{rcmh}}}[\pi(u)/\pi^{\mathbf{rcmh}}(u)] = \textstyle\sum_{u \in supp(\pi^{\mathbf{rcmh}})} \pi(u) \\ &= \textstyle\sum_{u \in supp(\pi)} \pi(u) = 1. \end{aligned}$$

Similarly, we have $\mathbb{E}_{p}[\pi(u)/p(u)] = 1$. Therefore, to prove the theorem, it is enough to show

$$\mathbb{E}_{\pi^{\mathbf{rcmh}}}[(\pi(u)/\pi^{\mathbf{rcmh}}(u))^2] \leq \mathbb{E}_{p}[(\pi(u)/p(u))^2].$$

Specifically, we have

$$\begin{aligned} &\mathbb{E}_{p}[(\pi(u)/p(u))^2] - \mathbb{E}_{\pi^{\mathbf{rcmh}}}[(\pi(u)/\pi^{\mathbf{rcmh}}(u))^2] \\ &= \sum_{u \in supp(p)} \pi^2(u)/p(u) - \sum_{u \in supp(\pi^{\mathbf{rcmh}})} \pi^2(u)/\pi^{\mathbf{rcmh}}(u) \\ &= \sum_{u \in supp(p)} \pi^2(u)[1/p(u) - 1/\pi^{\mathbf{rcmh}}(u)] \\ &= \sum_{u \in supp(p)} \pi^2(u)[\tfrac{\sum_{v \in supp(p)} p(v)}{p(u)} - \tfrac{\sum_{v \in supp(p)} \pi^{\mathbf{rcmh}}(v)}{\pi^{\mathbf{rcmh}}(u)}] \\ &= \sum_{u \in supp(p)} \sum_{v \in supp(p)} \pi^2(u)[\tfrac{p(v)}{p(u)} - \tfrac{\pi^{\mathbf{rcmh}}(v)}{\pi^{\mathbf{rcmh}}(u)}]. \end{aligned} \tag{6}$$

Then, by Theorem 3.1 and $p(u)P(u,v) = p(v)P(v,u)$, we have

$$\frac{\pi^{\mathbf{rcmh}}(v)}{\pi^{\mathbf{rcmh}}(u)} = \frac{p(v)(\pi(v)p(u))^{\alpha}}{p(u)(\pi(u)p(v))^{\alpha}}. \tag{7}$$

Algorithm 1 RCMH algorithm for graph sampling

1: $u \leftarrow$ initial node;
2: **while** stopping condition does not satisfy **do**
3: Select a node v uniformly at random from neighbors of u;
4: Generate a uniform random value $q \in [0, 1]$;
5: **if** $q \leq (d_u/d_v)^\alpha$ **then**
6: $u \leftarrow v$;
7: **else**
8: Stay at u;

Let $g(u, v) = \pi^2(u)[p(v)/p(u) - \pi^{\mathbf{rcmh}}(v)/\pi^{\mathbf{rcmh}}(u)]$. Then, for any pair of states $u, v \in supp(p)$, we let $h(u, v) = g(u, v) + g(v, u)$. To prove $\mathbb{E}_p[(\pi(u)/p(u))^2] - \mathbb{E}_{\pi^{\mathbf{rcmh}}}[(\pi(u)/\pi^{\mathbf{rcmh}}(u))^2] \geq 0$, it is sufficient to show $h(u, v) \geq 0$. Clearly, for $u = v$, we have $h(u, v) = 0$. For $u \neq v$, we have

$$\begin{aligned} h(u,v) &= \pi^2(u)\tfrac{p(v)}{p(u)}(1 - \tfrac{(\pi(v)p(u))^\alpha}{(\pi(u)p(v))^\alpha}) \\ &\quad + \pi^2(v)\tfrac{p(u)}{p(v)}(1 - \tfrac{(\pi(u)p(v))^\alpha}{(\pi(v)p(u))^\alpha}) \\ &= \pi(u)\pi(v)\tfrac{(\pi(u)p(v))}{(\pi(v)p(u))}[1 - \tfrac{(\pi(v)p(u))^\alpha}{(\pi(u)p(v))^\alpha}] \\ &\quad + \pi(u)\pi(v)\tfrac{(\pi(v)p(u))}{(\pi(u)p(v))}[1 - \tfrac{(\pi(u)p(v))^\alpha}{(\pi(v)p(u))^\alpha}]. \end{aligned} \tag{8}$$

Let $x = (\pi(u)p(v))/(\pi(v)p(u))$. Then, we have

$$\begin{aligned} h(u,v) &= \pi(u)\pi(v)[x(1 - 1/x^\alpha) + (1 - x^\alpha)/x] \\ &= \pi(u)\pi(v)[(x + 1/x) - (x^{1-\alpha} + 1/x^{1-\alpha})]. \end{aligned}$$

Let $g(x) = x + 1/x$. If $x \in (0, 1]$, we have $g'(x) = 1 - 1/x^2 \leq 0$, implying that $g(x)$ is monotonically decreasing. If $x \in [1, +\infty)$, we have $g'(x) \geq 0$, indicating that $g(x)$ is monotonically increasing. Now, we discuss the following two cases. If $x \in (0, 1]$, then $x \leq x^{1-\alpha}$ for $\alpha \in [0, 1]$. Since $g(x)$ is monotonically decreasing when $x \in (0, 1]$, we have $g(x) \geq g(x^{1-\alpha})$, resulting in $[(x + 1/x) - (x^{1-\alpha} + 1/x^{1-\alpha})] \geq 0$. Likewise, if $x \in [1, +\infty)$, then $x \geq x^{1-\alpha}$ holds. As a consequence, we have $g(x) \geq g(x^{1-\alpha})$ and $[(x+1/x)-(x^{1-\alpha}+1/x^{1-\alpha})] \geq 0$. Since $supp(p) = supp(\pi)$, we have $x > 0$ and $\pi(u)\pi(v) > 0$ for $u, v \in supp(p)$. Putting it all together, we have $h(u, v) \geq 0$. This completes the proof. ■

Note that Theorem 3.2 is very general. It does not require the explicit formula of $\pi^{\mathbf{rcmh}}$, and it only needs the knowledge of $\pi^{\mathbf{rcmh}}(v)/\pi^{\mathbf{rcmh}}(u)$ for any pair $u, v \in supp(\pi^{\mathbf{rcmh}})$ (see Eq. (6)).

B. RCMH algorithm for graph sampling

Here we show how to use the RCMH algorithm for unbiased graph sampling. In the case of unbiased graph sampling, the target distribution is $\pi = \pi^{\mathbf{u}}$. A random walk on graph G forms the initial Markov Chain P, where $P(u, v) = 1/d_u$ for all $v \in N(u)$ and $P(u, v) = 0$ for all $v \notin N(u)$. By Eq. (5), we can derive that the probability transition matrix of the RCMH algorithm for unbiased graph sampling is given by

$$P^{\mathbf{rcmh}}_{uv} = \begin{cases} (1/d_u)\min\{(d_u/d_v)^\alpha, 1\}, & if\ v \in N(u) \\ 1 - \sum_{u \neq w} P^{\mathbf{rcmh}}_{uw}, & if\ u = v, \\ 0, otherwise. \end{cases} \tag{9}$$

Algorithm 1 details the RCMH algorithm for unbiased graph sampling. Note that Algorithm 1 is nothing but a special case of the RCMH algorithm. After obtaining the samples by Algorithm 1, we are able to derive an asymptotically unbiased estimator for $\mathbb{E}_{\pi^{\mathbf{u}}}(f)$ based on the IS framework. First, we calculate the stationary distribution of RCMH algorithm for unbiased graph sampling. Specifically, based on Theorem 3.1, we have the following corollary.

Corollary 3.3: The stationary distribution of RCMH algorithm for unbiased graph sampling is given by $\pi^{\mathbf{rcmh}}(u) = d_u^{(1-\alpha)}/Z$ for $u \in V$, where Z is a normalization constant.

Proof: The proof is omitted due to space limit. ■

Equipped with Corollary 3.3, we present an estimator for the RCMH algorithm as follows:

$$\mathbb{E}_{\pi^{\mathbf{rcmh}}}(f) = \frac{\sum_{u \in S} f(u) w^{\mathbf{rcmh}}(u)}{\sum_{u \in S} w^{\mathbf{rcmh}}(u)}, \tag{10}$$

where S denotes the set of sampled nodes and $w^{\mathbf{rcmh}}(u) = d_u^{(\alpha-1)}$ is the important weight of node u. The asymptotical unbiasedness of the above estimator can be easily derived by a similar argument presented in [12], [2]. Note that by Theorem 3.2, we have $\mathrm{var}_{\pi^{\mathbf{rcmh}}}(\frac{\pi^{\mathbf{u}}(u)}{\pi^{\mathbf{rcmh}}(u)}) \leq \mathrm{var}_{\pi^{\mathbf{rw}}}(\frac{\pi^{\mathbf{u}}(u)}{\pi^{\mathbf{rw}}(u)})$. That is to say, Algorithm 1 can mitigate the *large deviation* problem of RW by setting an appropriate parameter α. On the other hand, Algorithm 1 can also alleviate the sample rejection problem of MH because its sample acceptance ratio is larger than that of MH given that $\alpha \in [0, 1)$. As a result, RCMH can achieve a tradeoff between the *large deviation* problem of RW and the sample rejection problem of MH. Additionally, when $\alpha = 1$, Algorithm 1 becomes the MH algorithm, and when $\alpha = 0$, Algorithm 1 becomes the RW algorithm. Therefore, Algorithm 1 creates an interesting connection between MH and RW, and it also unifies them.

IV. The GMD algorithm

In this section, we propose a novel generalized maximum-degree random walk (GMD) algorithm for unbiased graph sampling. We show that the GMD algorithm can balance the tradeoff between the *large deviation* problem of RW and the repeated samples problem of MD. Below, we first describe the basic idea of our algorithm. Then, we present the GMD algorithm and the theoretical analysis as well. Finally, we show how to use the GMD algorithm for unbiased graph sampling.

The basic idea. As discussed in Section II, MD needs to add self-loops on the nodes to achieve a uniform stationary distribution. If the maximum degree (or its upper bound) of a graph is very large[1], MD has to add a large number of self-loops on the low-degree nodes. As a result, the algorithm generates a large number of repeated samples. Intuitively, to mitigate the repeated samples problem of MD algorithm, one effective solution is to reduce the number of self-loops so that the walk goes to the neighbor nodes with a high probability. To achieve this end, our idea is to restrict the number of self-loops that are added on the low-degree nodes. In particular, we introduce a controlled parameter C into the original MD algorithm where C is a nonnegative integer. For any node

[1]This is indeed the case in the context of unbiased graph sampling via crawling. Because we do not know the maximum degree in advance, and thus we has to set a very large constant to ensure that it is larger than the maximum degree [7].

$u \in V$, if $d_u < C$, our algorithm adds $C - d_u$ self-loops on u, otherwise the algorithm does not add self-loops on u. Reconsider the example shown in Fig. 1. Fig. 1(a) and Fig. 1(c) depict the original graph and the modified graph produced by our algorithm given that $C = 3$. Note that when $C = 3$, we only need to add one self-loop each on nodes v_1 and v_4, because their degrees are smaller than C. For the nodes v_2, v_3 and v_5, we do not add self-loops as their degrees are no smaller than C. Compared to MD, our algorithm only adds two self-loops while MD has to add six self-loops. Therefore, the number of repeated samples produced by our algorithm is expected to be substantially smaller than the number of repeated samples generated by MD.

Similar to MD [7], the procedure of adding self-loops in GMD can also be done implicitly and on-the-fly. Specifically, when the GMD walk (with parameter C) traverses a node u, the walk picks a neighbor node from $N(u)$ with probability $1/\max\{d_u, C\}$. According to this process, for the node whose degree is no less than C, the GMD walk equals the traditional random walk to pick the next node. For the node whose degree is smaller than C, then with probability $1/C$ the GMD walk selects the next node from $N(u)$, and with probability $(C - d_u)/C$ stays at u. This is equivalent to the process of adding $C - d_u$ self-loops on u and then performing the traditional random walk on the modified graph to select the next node.

A. GMD algorithm and theoretical analysis

Based on the above idea, the transition probability of the GMD random walk is defined as

$$P^{\mathbf{gmd}}(u,v) = \begin{cases} 1/\max\{d_u, C\}, & if\ v \in N(u) \\ 0, & otherwise, \end{cases} \tag{11}$$

where C is a nonnegative integer. Notice that when $C = 0$, $P^{\mathbf{gmd}}(u,v) = 1/d_u$ for $v \in N(u)$, and thus the GMD random walk becomes the traditional random walk. When $C = d_{\max}$, $P^{\mathbf{gmd}}(u,v) = 1/d_{\max}$ for $v \in N(u)$, and thereby the GMD random walk becomes the maximum-degree random walk. Let $P^{\mathbf{md}}(u,v)$ be the transition probability of the maximum-degree random walk. Then, by our definition, $P^{\mathbf{gmd}}(u,v) \geq P^{\mathbf{md}}(u,v)$ for any u, v. This fact indicates that at any node $u \in V$, the probability of the GMD walk traversing the self-loops is smaller than the probability of the maximum-degree walk traversing the self-loops. Below, we prove that the expected "self-loops-traversed" ratio of the GMD walk is smaller than that of the maximum-degree walk given that both the GMD walk and the maximum-degree walk have converged into their stationary distributions. Before we proceed, we first derive the stationary distribution of the GMD random walk denoted by $\pi^{\mathbf{gmd}}$ as follows:

$$\pi^{\mathbf{gmd}}(u) = \max\{d_u, C\}/\sum\nolimits_{u\in V} \max\{d_u, C\}, \tag{12}$$

for any $u \in V$. Denote by $r^{\mathbf{gmd}}(u)$ and by $r^{\mathbf{md}}(u)$ the probabilities of the GMD walk and the maximum-degree walk traversing the self-loops in the next step, respectively. By definition, it follows that $r^{\mathbf{gmd}}(u) = 1 - d_u/(\max\{d_u, C\})$ and $r^{\mathbf{md}}(u) = 1 - d_u/d_{\max}$. The expected "self-loops-traversed" ratio of the GMD walk and of the maximum-degree walk are defined by $\mathbb{E}_{\pi^{\mathbf{gmd}}}[r^{\mathbf{gmd}}(u)]$ and $\mathbb{E}_{\pi^{\mathbf{md}}}[r^{\mathbf{md}}(u)]$ respectively. The following theorem shows that $\mathbb{E}_{\pi^{\mathbf{gmd}}}[r^{\mathbf{gmd}}(u)] < \mathbb{E}_{\pi^{\mathbf{md}}}[r^{\mathbf{md}}(u)]$, given that $C < d_{\max}$.

Theorem 4.1: $\mathbb{E}_{\pi^{\mathbf{gmd}}}[r^{\mathbf{gmd}}(u)] < \mathbb{E}_{\pi^{\mathbf{md}}}[r^{\mathbf{md}}(u)]$, given that $C < d_{\max}$.

Proof: By our definition and Eq. (12), we have

$$\begin{aligned} &\mathbb{E}_{\pi^{\mathbf{gmd}}}[r^{\mathbf{gmd}}(u)] = \sum_{u\in V} \pi^{\mathbf{gmd}}(u) r^{\mathbf{gmd}}(u) \\ &= 1 - \sum_{u\in V} d_u / \sum_{u\in V} \max\{d_u, C\} < 1 - \sum_{u\in V} d_u / \sum_{u\in V} d_{\max} \\ &= \mathbb{E}_{\pi^{\mathbf{md}}}[r^{\mathbf{md}}(u)]. \end{aligned}$$

This completes the proof. ∎

Theorem 4.1 implies that if $C < d_{\max}$, the number of repeated samples obtained by the GMD algorithm is expected to be smaller than the number of repeated samples generated by MD, as the expected "self-loops-traversed" ratio of the GMD walk is smaller than that of the maximum-degree walk. The GMD algorithm therefore alleviates the repeated samples problem of MD algorithm provided that $C < d_{\max}$. In the experiments, we shall show that the GMD algorithm decreases the number of repeated samples compared to MD. Next, we prove that GMD algorithm is also able to mitigate the *large deviation* problem of RW. Specifically, the following theorem shows that the GMD algorithm reduces the χ-distance between the target distribution $\pi^{\mathbf{u}}$ and the trial distribution $\pi^{\mathbf{rw}}$ of RW.

Theorem 4.2: $\mathrm{var}_{\pi^{\mathbf{gmd}}}[\frac{\pi^{\mathbf{u}}(u)}{\pi^{\mathbf{gmd}}(u)}] \leq \mathrm{var}_{\pi^{\mathbf{rw}}}[\frac{\pi^{\mathbf{u}}(u)}{\pi^{\mathbf{rw}}(u)}]$

Proof: First, we have $\mathbb{E}_{\pi^{\mathbf{gmd}}}[\frac{\pi^{\mathbf{u}}(u)}{\pi^{\mathbf{gmd}}(u)}] = \sum_{u\in V} \pi^{\mathbf{gmd}}(u) \frac{\pi^{\mathbf{u}}(u)}{\pi^{\mathbf{gmd}}(u)} = 1$, and $\mathbb{E}_{\pi^{\mathbf{rw}}}[\frac{\pi^{\mathbf{u}}(u)}{\pi^{\mathbf{rw}}(u)}] = \sum_{u\in V} \pi^{\mathbf{gmd}}(u) \frac{\pi^{\mathbf{u}}(u)}{\pi^{\mathbf{gmd}}(u)} = 1$. Then, to prove the theorem, it is sufficient to show that

$$\mathbb{E}_{\pi^{\mathbf{gmd}}}[(\frac{\pi^{\mathbf{u}}(u)}{\pi^{\mathbf{gmd}}(u)})^2] \leq \mathbb{E}_{\pi^{\mathbf{rw}}}[(\frac{\pi^{\mathbf{u}}(u)}{\pi^{\mathbf{rw}}(u)})^2].$$

In particular, we have

$$\begin{aligned} &\mathbb{E}_{\pi^{\mathbf{gmd}}}[(\tfrac{\pi^{\mathbf{u}}(u)}{\pi^{\mathbf{gmd}}(u)})^2] - \mathbb{E}_{\pi^{\mathbf{rw}}}[(\tfrac{\pi^{\mathbf{u}}(u)}{\pi^{\mathbf{rw}}(u)})^2] \\ &= \sum_{u\in V} [\tfrac{(\pi^{\mathbf{u}}(u))^2}{\pi^{\mathbf{gmd}}(u)} - \tfrac{(\pi^{\mathbf{u}}(u))^2}{\pi^{\mathbf{rw}}(u)}] = \tfrac{1}{n^2} \sum_{u\in V} [\tfrac{1}{\pi^{\mathbf{gmd}}(u)} - \tfrac{1}{\pi^{\mathbf{rw}}(u)}] \\ &= \tfrac{1}{n^2} \sum_{u\in V} [\tfrac{\sum_{v\in V} \pi^{\mathbf{gmd}}(v)}{\pi^{\mathbf{gmd}}(u)} - \tfrac{\sum_{v\in V} \pi^{\mathbf{rw}}(v)}{\pi^{\mathbf{rw}}(u)}] \\ &= \tfrac{1}{n^2} \sum_{u\in V} \sum_{v\in V} [\tfrac{\pi^{\mathbf{gmd}}(v)}{\pi^{\mathbf{gmd}}(u)} - \tfrac{\pi^{\mathbf{rw}}(v)}{\pi^{\mathbf{rw}}(u)}]. \end{aligned} \tag{13}$$

By definition and Eq. (12), we have $\pi^{\mathbf{rw}}(v)/\pi^{\mathbf{rw}}(u) = d_v/d_u$ and $\pi^{\mathbf{gmd}}(v)/\pi^{\mathbf{gmd}}(u) = \max\{d_v, C\}/\max\{d_u, C\}$. Denote by $g(u,v) = \pi^{\mathbf{gmd}}(v)/\pi^{\mathbf{gmd}}(u) - \pi^{\mathbf{rw}}(v)/\pi^{\mathbf{rw}}(u)$. Then, for any pair of nodes $u, v \in V$, we let $h(u,v) = g(u,v) + g(v,u)$. Based on Eq. (13), to prove $\mathbb{E}_{\pi^{\mathbf{gmd}}}[(\frac{\pi^{\mathbf{u}}(u)}{\pi^{\mathbf{gmd}}(u)})^2] \leq \mathbb{E}_{\pi^{\mathbf{rw}}}[(\frac{\pi^{\mathbf{u}}(u)}{\pi^{\mathbf{rw}}(u)})^2]$, it suffices to prove $h(u,v) \leq 0$. First, for $u = v$, we have $h(u,v) = 0$. Second, for $u \neq v$, we have $h(u,v) = \frac{\max\{d_v,C\}}{\max\{d_u,C\}} - \frac{d_v}{d_u} + \frac{\max\{d_u,C\}}{\max\{d_v,C\}} - \frac{d_u}{d_v}$. Without loss of generality, we assume that $d_u \geq d_v$. Then, we consider the following three cases. (1) If $d_u \geq d_v \geq C$, we have $h(u,v) = 0$. (2) If $d_u \geq C \geq d_v$, we have $h(u,v) = \frac{C}{d_u} - \frac{d_v}{d_u} + \frac{d_u}{C} - \frac{d_u}{d_v} = \frac{(C-d_v)(Cd_v - d_u^2)}{Cd_u d_v} \leq 0$. (3) If $C \geq d_u \geq d_v$, we have $h(u,v) = 1 - \frac{d_v}{d_u} + 1 - \frac{d_u}{d_v} \leq 0$. Putting it all together, we conclude that $h(u,v) \leq 0$. Thus, the theorem is established. ∎

Algorithm 2 GMD algorithm for graph sampling

1: $u \leftarrow$ initial node;
2: $i \leftarrow 1$;
3: **while** stopping condition does not meet **do**
4: $\quad \xi_i \leftarrow$ Geometric($d_u / \max\{d_u, C\}$); $S_i \leftarrow u$;
5: $\quad$ Select a node v uniformly at random from $N(u)$;
6: $\quad u \leftarrow v$; $i \leftarrow i + 1$;
7: **return** S and ξ;

B. GMD algorithm for graph sampling

Here we show how to use the GMD algorithm for unbiased graph sampling. To this end, we first perform a GMD random walk on a graph to collect nodes. In each step, the GMD walk collects one node. Note that by this procedure, when the walk traverses the self-loops, we need to record the repeated nodes into the sample set. This method, however, is not very effective because the walk could traverse the self-loops too many times when it arrives at a low-degree node. Similar to the optimized MD algorithm presented in [7], [11], we can use a geometric random variable to model the number of consecutive self-loops that the GMD walk takes on each node so that one computation can simulate many steps of the actual walk. More specifically, when the GMD walk reaches a node u, we generate a geometric random variable $\xi(u)$ with the success probability $d_u / \max\{d_u, C\}$ to simulate that the walk traverses the self-loops. After that, the GMD walk picks the next node from $N(u)$ with probability $1/d_u$. The detailed description of the GMD algorithm is given in Algorithm 2. Note that in line 4 of Algorithm 2, ξ_i is a geometric random variable with success probability $d_u / \max\{d_u, C\}$. Algorithm 2 outputs the collected node-set S and the corresponding set of geometric random variables ξ.

After obtaining S and ξ by Algorithm 2, we can construct an estimator for $\mathbb{E}_{\pi^{\mathbf{u}}}(f)$ based on the IS framework [12]. In particular, the estimator in the GMD algorithm is given by

$$\mathbb{E}_{\pi^{\mathbf{gmd}}}(f) = \frac{\sum_{i=1}^{|S|} f(S_i)\xi_i / \max\{d_{S_i}, C\}}{\sum_{i=1}^{|S|} \xi_i / \max\{d_{S_i}, C\}}, \tag{14}$$

where S_i denotes the i-th node collected by Algorithm 2, and ξ_i be the corresponding geometric random variable denoting the multiplicity of the sample S_i. One can use a similar argument presented in [12], [2] to prove that the estimator $\mathbb{E}_{\pi^{\mathbf{gmd}}}(f)$ is asymptotically unbiased. Notice that when $C = 0$, the success probability of the geometric random variable is 1; thus $\xi_i = 1$ for any $i = 1, \cdots, |S|$. In this case, the estimator $\mathbb{E}_{\pi^{\mathbf{gmd}}}(f)$ becomes the estimator $\mathbb{E}_{\pi^{\mathbf{rw}}}(f)$ (Eq. (2)). When $C = d_{\max}$, the estimator becomes the estimator used in the MD algorithm [7]. Thus, our estimator can be deemed as a generalized estimator which treats the estimators used in RW and MD as two special cases.

Discussion. Compared to MD, the GMD algorithm has two advantages. First, GMD is more flexible and does not require the knowledge of maximum degree (or guessing an upper bound). Second, by Theorem 4.1, GMD can alleviate the repeated samples problem of MD with a cost of increasing the deviation between the target and trial distributions. In the experiments, however, we will show that such a cost is typically very low in many real-world graphs. Compared with the RW algorithm, our algorithm can mitigate the *large deviation* problem (Theorem 4.2) with an expense of increasing repeated samples. In the experiments, we find that the proposed algorithm can provide a good tradeoff between the repeated samples problem of MD and the *large deviation* problem of RW by setting an appropriate parameter C. In addition, our algorithm is very general, treating both the MD and RW as two special instances. In this sense, the proposed algorithm establishes an interesting link between RW and MD, and it also unifies these two algorithms.

V. Optimized algorithms

In this section, we present several optimization techniques to further improve the estimation accuracy of the RCMH and GMD algorithm. Specifically, we leverage the delayed acceptance and non-backtracking walk techniques proposed in [2] to optimize the RCMH and GMD algorithms, respectively. The resulting optimized algorithms are shown to have a smaller asymptotic variance than their original counterparts. In the following, we detail the optimized RCMH and GMD algorithms respectively.

A. RCMH with delayed acceptance

Here we show how the RCMH algorithm can be integrated with the delayed acceptance technique [2] to further improve the estimation accuracy. The delayed acceptance technique is a powerful technique to improve the performance of the Metropolis-Hastings algorithms. This technique makes use of the idea of non-backtracking walk which does not affect the stationary distribution of the original Metropolis-Hastings algorithm. Moreover, it has shown that it reduces the asymptotic variance of the original estimator obtained by the Metropolis-Hastings algorithm. Due to space limit, we only describe the idea of the delayed acceptance technique and how it is integrated into our algorithm. Let w and u be the previous and current states of the random walk of the Metropolis-Hastings algorithm, respectively. Then, if the next state, say v, is accepted by Metropolis-Hastings algorithm, then the delayed acceptance technique needs to consider the following two scenarios. First, if v is equal to the previous state w, then the delayed acceptance technique makes the walk stop going to v (i.e., the acceptance is delayed). Instead, it proposes to go to another state $v' \in \mathcal{U} \backslash \{v\}$ using another Metropolis-Hastings algorithm (more details can be found in their original paper [2]). Second, if $v \neq w$, then the delayed acceptance technique makes the walk go to v like the traditional Metropolis-Hastings algorithm.

Likewise, for the RCMH algorithm, we can also use such powerful delayed acceptance technique. Unlike the traditional Metropolis-Hastings algorithm, our acceptance function is given in Eq. (4). When a state is accepted, then we apply the delayed acceptance technique to avoid the walk to backtrack to the previous state as the traditional Metropolis-Hastings algorithm with delayed acceptance does. One can make use of the similar arguments to show that such a modification does not affect the stationary distribution of the RCMH algorithm. For convenience, we refer to this modified RCMH algorithm as the RCMH with delayed acceptance algorithm, and it is abbreviated by RCMHDA. Thus, after obtaining the samples by the RCMHDA algorithm, we use the same estimator as

stated in Eq. (10). Using a similar proof presented in [2], we can conclude that this estimator cuts the asymptotic variance of the estimator obtained by the RCMH algorithm.

B. Non-backtracking GMD algorithm

Here we propose a non-backtracking GMD (NGMD) algorithm for unbiased graph sampling which integrates the idea of non-backtracking random walk [2] into the GMD algorithm. We show that the NGMD algorithm not only preserves the same stationary distribution as that of the GMD algorithm, but it also reduces the asymptotic variance [2], [12] of GMD.

First, let us introduce some important notations and concepts. Denote by $X_t \in V$, for $t = 0, 1, \cdots$, the location of a random walk on an undirected graph at discrete time t, and by $P(X_t, X_{t+1})$ the transition probability. Then, an augmented state space of this random walk is defined by

$$\mathcal{E} \triangleq \{(u, v) | u, v \in V, P(u, v) > 0\} \subseteq V \times V. \tag{15}$$

Denote by e_{uv} a state $(u, v) \in \mathcal{E}$. Note that by this definition, $e_{uv} \neq e_{vu}$, and it is not necessary $u \neq v$. For instance, in the case of GMD random walk, it exists self-loops, i.e., $P(u, u) > 0$ for some u. Thus, by Eq. (15), it is possible that $e_{uu} \in \mathcal{E}$ for some u. It is well-known that on the augmented state space $\mathcal{E}$ the random walk still forms a Markov Chain denoted by $\{Y_t \triangleq (X_{t-1}, X_t)\}$ [22], [28], [2]. Let $Q(e_{uv}, e_{wz})$ be the transition matrix of such a Markov Chain on $\mathcal{E}$. Then, by the definition of random walk, we have $Q(e_{uv}, e_{wz}) = 0$ when $v \neq w$. Denote by φ the stationary distribution of the Markov Chain on $\mathcal{E}$, and by π the stationary distribution of the random walk on the state space V. If $\varphi(e_{uv}) = \pi(u)P(u, v)$ for all $e_{uv} \in \mathcal{E}$, then we have

$$\sum\nolimits_{u \in V, e_{uv} \in \mathcal{E}} \varphi(e_{uv}) = \sum\nolimits_{u \in V} \pi(u) P(u, v) = \pi(v), \tag{16}$$

where the first equality holds due to $P(u, v) = 0$ for $e_{uv} \notin \mathcal{E}$ [2]. Next, we can define a new function $F : \mathcal{E} \to \mathbb{R}$ such that $F(e_{uv}) = f(v)$. Then, we have

$$\begin{aligned} &\mathbb{E}_\varphi(F) = \sum_{e_{uv} \in \mathcal{E}} \varphi(e_{uv}) F(e_{uv}) = \sum_{u, v \in V} \pi(u) P(u, v) f(v) \\ &= \textstyle\sum_{v \in V} \pi(v) f(v) = \mathbb{E}_\pi(f). \end{aligned} \tag{17}$$

Let $\hat{\mathbb{E}}_\varphi(F)$ be an unbiased estimator of $\mathbb{E}_\varphi(F)$ constructed from the Markov Chain $\{Y_t\}$ on the augmented state space $\mathcal{E}$, and $\hat{\mathbb{E}}_\pi(f)$ be an unbiased estimator of $\mathbb{E}_\pi(f)$ obtained from the random walk $\{X_t\}$ on the original state space V. Obviously, by Eq. (17), $\hat{\mathbb{E}}_\varphi(F)$ is also unbiased for $\mathbb{E}_\pi(f)$. There are many different constructions of the Markov Chain $\{Y_t\}$ depending on different transition matrices Q. For some constructions, the asymptotic variance of the estimator $\hat{\mathbb{E}}_\varphi(F)$ is no larger than that of the estimator $\hat{\mathbb{E}}_\pi(f)$ [28], [2]. Note that Eq. (17) is subject to the condition $\varphi(e_{uv}) = \pi(u)P(u, v)$. Therefore, an important question is how to construct a Markov Chain $\{Y_t\}$ such that the asymptotic variance of $\hat{\mathbb{E}}_\varphi(F)$ is no greater than that of $\hat{\mathbb{E}}_\pi(f)$, while simultaneously maintaining the condition $\varphi(e_{uv}) = \pi(u)P(u, v)$. Neal in [28] proposed a general method to achieve this goal. Specifically, he proved the following theorem.

Theorem 5.1: **[28]** Let $\{X_t\}$ be an irreducible and reversible Markov Chain with transition matrix P and stationary distribution π. Denote by $\{Y_t \triangleq (X_{t-1}, X_t)\}$ a new Markov Chain defined on the augmented space $\mathcal{E}$ with transition matrix Q. If Q meets the following two conditions:

$$P(u, w)Q(e_{wu}, e_{uv}) = P(u, v)Q(e_{vu}, e_{uw}), \tag{18}$$

$$Q(e_{wu}, e_{uv}) \geq P(u, v), \tag{19}$$

for all $e_{wu}, e_{uw}, e_{uv}, e_{vu} \in \mathcal{E}$ with $w \neq v$. Then, the Markov Chain $\{Y_t\}$ has a unique stationary distribution φ where $\varphi(e_{uv}) = \pi(u)P(u, v)$ holds for any $e_{uv} \in \mathcal{E}$. Moreover, for any function f, the asymptotic variance of $\hat{\mathbb{E}}_\varphi(F)$ is no larger than that of $\hat{\mathbb{E}}_\pi(f)$.

Non-backtracking random walk with re-weighting (NRW). Equipped with Theorem 5.1, the challenge left is how to construct the transition matrix Q that meets the conditions shown in Eq. (18) and Eq. (19). In [2], Lee et al. proposed a non-backtracking random walk algorithm (NRW) for unbiased graph sampling where the transition matrix satisfies the conditions in Theorem 5.1. The NRW algorithm first performs a non-backtracking random walk to collect nodes, and then uses the same estimator as that of the RW algorithm (Eq. (2)) to estimate $\mathbb{E}_{\pi^{\mathrm{u}}}(f)$. Here the non-backtracking random walk is a variant of the traditional random walk in which the walk does not backtrack to the previous node. Specifically, for any $u, v \in V$, let $P^{\mathbf{rw}}(u, v) = 1/d_u$ and $P^{\mathbf{nrw}}(u, v)$ be the transition probabilities of the traditional random walk and the non-backtracking random walk, respectively. Here $P^{\mathbf{nrw}}(u, v) = 1/(d_u - 1)$ if $d_u > 1$; otherwise $P^{\mathbf{nrw}}(u, v) = 1/d_u$. Note that $P^{\mathbf{nrw}}(u, v) = 1/(d_u - 1)$ (not $1/d_u$) for $d_u > 1$ implies that the walk does not backtrack to the previous node where the walk originates. This non-backtracking random walk can be interpreted by a Markov Chain defined on the augmented space $\mathcal{E}$. In particular, let w, u, and v be the previous state, the current state, and the next state of a traditional random walk on graph, respectively. Then, for all $w \neq v$ and $e_{wu}, e_{uv} \in \mathcal{E}$, we can construct a Markov Chain on the augmented space $\mathcal{E}$ with transition probabilities $Q(e_{wu}, e_{uv}) = 1/(d_u - 1)$ for $d_u > 1$ and $Q(e_{wu}, e_{uv}) = 1/d_u$ for $d_u = 1$. By this construction, one can show that Q satisfies the conditions in Theorem 5.1 [2]. Clearly, the transition matrix of the non-backtracking random walk $P^{\mathbf{nrw}}$ is equivalent to Q. By Theorem 5.1 and Eq. (16), the stationary distribution (w.r.t. nodes) of the non-backtracking random walk is still the same as that of the traditional random walk. As a result, after obtaining the sampled nodes by the non-backtracking random walk, we can use the same estimator as that of the RW algorithm (Eq. (2)). Notice that by Theorem 5.1, the estimator obtained by the non-backtracking random walk has a smaller asymptotic variance than the estimator obtained by the traditional random walk.

The NGMD algorithm. Here we propose a novel non-backtracking GMD random walk algorithm called NGMD which applies the non-backtracking random walk technique to the GMD algorithm. It is important to note that this is non-trivial because the GMD random walk may traverse self-loops, while the traditional random walk does not. Hence, the idea of the NRW algorithm cannot be directly applied in our case. There are two challenges to devise a non-backtracking GMD random walk algorithm. The first challenge is how to design a transition matrix Q for the Markov Chain defined on the

augmented state space $\mathcal{E}$ such that it satisfies the conditions in Theorem 5.1 given that the original random walk is the GMD walk. The second challenge is how to devise an efficient NGMD algorithm using the geometric random variable to model the number of consecutive self-loops subject to the constraint of non-backtracking walk. Below, we shall present a new algorithm to tackle these challenges.

Let w, u, and v be the previous state, the current state, and the next state of a GMD random walk on a graph, respectively. Based on the GMD random walk, we construct a Markov Chain on the augmented space $\mathcal{E}^{\mathbf{ngmd}}$ with transition matrix $Q^{\mathbf{ngmd}}$ where $\mathcal{E}^{\mathbf{ngmd}} \triangleq \{(u,v)|u,v \in V, P^{\mathbf{gmd}}(u,v) > 0\}$. Denote by s_u the number of self-loops that are added on u by the GMD algorithm, i.e., $s_u = \max\{d_u, C\} - d_u$. For any $e_{uv}, e_{wz} \in \mathcal{E}^{\mathbf{ngmd}}$, if $v \neq w$, we define $Q^{\mathbf{ngmd}}(e_{uv}, e_{wz}) = 0$. Otherwise, we define the $Q^{\mathbf{ngmd}}$ as follows.

For any $e_{wu}, e_{uv} \in \mathcal{E}^{\mathbf{ngmd}}$, if $\max\{d_u, C\} = 1$, we define $Q^{\mathbf{ngmd}}(e_{wu}, e_{uv}) = 1$. Unlike NRW, if $\max\{d_u, C\} > 1$, we have to consider two different cases. First, when $w = u$, implying that the GMD walk traverses a self-loop $u \rightarrow u$, we define the transition probabilities as

$$Q^{\mathbf{ngmd}}(e_{uu}, e_{uv}) = \begin{cases} 1/(\max\{d_u, C\} - 1), \ if \ u \neq v, \\ (s_u - 1)/(\max\{d_u, C\} - 1), \ if \ u = v \end{cases} \tag{20}$$

where $e_{uu}, e_{uv} \in \mathcal{E}^{\mathbf{ngmd}}$. The idea behind the above equation is as follows. Recall that under the case of $w = u$, the GMD walk comes from node w, i.e., the previous state is w. Therefore, when the GMD walk traverses the neighbor node of u in the next step (the case $u \neq v$ in Eq. (20)), the walk clearly never backtracks to w. However, when the GMD walk traverses a self-loop of u in the next step (the case $u = v$ in Eq. (20)), the walk should avoid to traverse "the same self-loop" where the walk originates, i.e., w. That is to say, we should "reduce" one self-loop on u. As a result, the transition probability is given by $(s_u - 1)/(\max\{d_u, C\} - 1)$. One can easily check that under the case of $w = u$, $Q^{\mathbf{ngmd}}(e_{uu}, e_{zv})$ is a valid transition probability because $\sum_{e_{zv} \in \mathcal{E}^{\mathbf{ngmd}}} Q^{\mathbf{ngmd}}(e_{uu}, e_{zv}) = 1$.

Second, when $w \neq u$, indicating that the GMD walk does not traverse a self-loop in the current step, we define the transition probabilities as

$$Q^{\mathbf{ngmd}}(e_{wu}, e_{uv}) = \begin{cases} s_u/(\max\{d_u, C\} - 1), \ if \ u = v, \\ 1/(\max\{d_u, C\} - 1), \ if \ u \neq v \wedge w \neq v, \\ 0, \ otherwise, \end{cases} \tag{21}$$

where $e_{wu}, e_{uv} \in \mathcal{E}^{\mathbf{ngmd}}$. We describe the idea behind Eq. (21) as follows. When the GMD walk traverses the self-loops in the next step (the case $u = v$ in Eq. (21)), the walk will not backtrack to the previous state w as $w \neq u$. Therefore, the probability of the walk traversing the self-loops is $s_u/(\max\{d_u, C\} - 1)$ as there are s_u self-loops. When the GMD walk traverses the neighbor node of u in the next step (the case $u \neq v$), the walk should avoid to go to the previous node w. Thus, the transition probability is $1/(\max\{d_u, C\} - 1)$ if $w \neq v$, and otherwise the transition probability is 0. One can easily show that under the case of $w \neq u$, $Q^{\mathbf{ngmd}}(e_{wu}, e_{zv})$ is a valid transition probability. More importantly, by these constructions, $Q^{\mathbf{ngmd}}$ meets the conditions in Theorem 5.1 (Eq. (18) and Eq. (19)) as stated in the following theorem.

Theorem 5.2: $Q^{\mathbf{ngmd}}$ meets the conditions in Theorem 5.1.

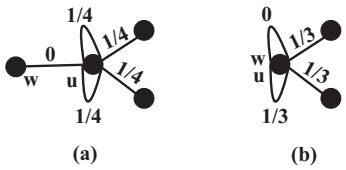

Fig. 2: Illustration of NGMD walk's transition probability

Proof: If $\max\{d_u, C\} = 1$, by definition, one can easily verify that the theorem holds. If $\max\{d_u, C\} > 1$, we consider the following three different cases. (1) $w = u$ and $u \neq v$. In this case, by definition, we have $P^{\mathbf{gmd}}(u,w) = s_u/\max\{d_u, C\}$, $P^{\mathbf{gmd}}(u,v) = 1/\max\{d_u, C\}$, $Q^{\mathbf{ngmd}}(e_{wu}, e_{uv}) = 1/(\max\{d_u, C\} - 1)$, and $Q^{\mathbf{ngmd}}(e_{vu}, e_{uw}) = s_u/(\max\{d_u, C\} - 1)$. Based on this, one can easily check that the first condition in Theorem 5.1, i.e., Eq. (18), holds. Also, we have $Q^{\mathbf{ngmd}}(e_{wu}, e_{uv}) > P^{\mathbf{gmd}}(u,v)$. (2) $w \neq u$ and $u = v$. In this case, we have $P^{\mathbf{gmd}}(u,w) = 1/\max\{d_u, C\}$, $P^{\mathbf{gmd}}(u,v) = s_u/\max\{d_u, C\}$, $Q^{\mathbf{ngmd}}(e_{wu}, e_{uv}) = (s_u)/(\max\{d_u, C\} - 1)$, and $Q^{\mathbf{ngmd}}(e_{vu}, e_{uw}) = 1/(\max\{d_u, C\} - 1)$. Also, we can conclude that under this case, $P^{\mathbf{gmd}}$ and $Q^{\mathbf{ngmd}}$ meet the conditions in Theorem 5.1. (3) $w \neq u$, $u \neq v$, and $w \neq v$. In this case, by definition, we have $P^{\mathbf{gmd}}(u,w) = 1/\max\{d_u, C\}$, $P^{\mathbf{gmd}}(u,v) = 1/\max\{d_u, C\}$, $Q^{\mathbf{ngmd}}(e_{wu}, e_{uv}) = 1/(\max\{d_u, C\} - 1)$, and $Q^{\mathbf{ngmd}}(e_{vu}, e_{uw}) = 1/(\max\{d_u, C\} - 1)$. Similarly, we can claim that $P^{\mathbf{gmd}}$ and $Q^{\mathbf{ngmd}}$ satisfy the conditions in Theorem 5.1. It is worth noting that we do not need to consider the case of $w = v$ because the conditions in Theorem 5.1 requires $w \neq v$. Putting it all together, the theorem is established. ■

Based on Eq. (20) and Eq. (21), for any w, u, v, the transition probability of the NGMD random walk is defined by

$$P^{\mathbf{ngmd}}(u,v) = Q^{\mathbf{ngmd}}(e_{wu}, e_{uv}). \tag{22}$$

Compared to the GMD random walk, the next step of the NGMD random walk depends on both the current node and the previous node. Therefore, the NGMD walk has to record the previous node. It is important to note that in the context of graph sampling via crawling, the primary cost is the number of samples that the random walk needs to draw [1], [2]. Hence, such an additional space overhead in the NGMD walk does not matter in this context. Fig. 2 illustrates the transition probability of the NGMD walk. In Fig. 2(a), the NGMD walk is currently at node u and the previous state is w where $w \neq u$. According to Eq. (21) and Eq. (22), the walk with probability 0 backtracks to w and with equal probability (1/4 in Fig. 2(a)) goes to the other neighbor nodes and the self-loops. Similarly, if the previous node w equals u (Fig. 2(b)) which means that the walk has traversed a self-loop, then the walk with probability 0 traverses the same self-loop and with the equal probability (1/3 in Fig. 2(b)) traverses the other self-loops and the neighbor nodes. Note that by Theorem 5.1, Theorem 5.2, and Eq. (16), the NGMD random walk achieves the same stationary distribution as that of the GMD random walk.

Now we turn to overcome the second challenge, i.e., how to optimize the NGMD algorithm using the geometric random variable model. Notice that there are two different cases that the NGMD walk may traverse self-loops consecutively: (1)

TABLE I: Summary of the datasets

Name	# of nodes	# of edges	TVD	$d_{\max}$
Digg	116,893	14,523,048	0.727	6,957
WikiTalk	2,363,335	8,223,638	0.603	95,354
WebStanford	157,144	681,398	0.394	237
RoadnetCA	1,965,206	5,533,214	0.136	12

$w = u$ (the previous state of the NGMD walk equals the current state), and (2) $w \neq u$. A straightforward solution is to use two geometric random variables to simulate the consecutive self-loops in these two cases, respectively. Unfortunately, this is not correct. The reason is because under case (2), if the NGMD walk traverses a self-loop, then in the next step case (2) becomes case (1) (because in the next step the previous state and the current state are the same). Therefore, under case (2), we cannot use a geometric random variable to simulate the consecutive self-loops. In fact, we find that we only need to adopt the geometric random variable model under case (1). Specifically, when $w = u$, according to Eq. (20), we generate a geometric random variable $\xi(u)$ with success probability $d_u/(\max\{d_u, C\} - 1)$ simulating the number of consecutive self-loops that the NGMD walk takes at u. When $w \neq u$, we use the transition probability defined in Eq. (21) to select the next node. If the next node is u (the current state), i.e., the NGMD walk traverses the self-loop, then we turn to case (1). If the next node is not u, then we perform the same procedure recursively. The detailed algorithm of NGMD is outlined in Algorithm 3. In Algorithm 3, lines 4-8 and lines 10-16 implement the NGMD random walk under case (1) and case (2), respectively. Note that in line 5, ξ_i denotes the multiplicity of the sample S_i. This quantity equals the geometric random variable with success probability $d_u/(\max\{d_u, C\} - 1)$ plus 1 because we need to add the self-loop $w \to u$ given that $w = u$. Line 20 means that the NGMD walk traverses the self-loop under case (2). After that, the random walk turns to case (1). The algorithm returns the sample set S and the vector ξ. After obtaining S and ξ, we can use the same estimator defined in Eq. (14) to estimate $\mathbb{E}_{\pi^{\mathrm{u}}}(f)$ as the stationary distribution of the NGMD walk is equal to that of the GMD walk. By Theorem 5.1 and Theorem 5.2, we can conclude that the estimator obtained by the NGMD random walk has smaller asymptotic variance than the estimator obtained by the GMD random walk.

Additionally, we remark that in Algorithm 3, if $C = 0$, Algorithm 3 becomes the NRW algorithm presented in [2]. This is because if $C = 0$, the case of $w = u$ (line 4-8 in Algorithm 3) is impossible and the condition in line 11 always holds. If $C = d_{\max}$, we refer to the Algorithm 3 as the non-backtracking maximum-degree random walk (NMD) algorithm. In this sense, similar to the GMD algorithm, the NGMD algorithm (Algorithm 3) is also very general and treats NRW and NMD algorithms as two special instances.

Algorithm 3 NGMD algorithm for graph sampling

```
1: w ← previous node, u ← current node;
2: i ← 1;
3: while stopping condition does not meet do
4:    if w = u then
5:       ξ_i ← Geometric(d_u/(max{d_u, C} − 1)) + 1;
6:       S_i ← u;
7:       Select a node v uniformly at random from N(u);
8:       u ← v; i ← i + 1;
9:    else
10:      Generate a uniform random value p ∈ [0, 1];
11:      if p × (max{d_u, C} − 1) ≤ d_u − 1 then
12:         ξ_i ← 1; S_i ← u;
13:         Select a node v uniformly at random from N(u)\{w};
14:         w ← u; u ← v; i ← i + 1;
15:      else
16:         w ← u;
17: return S and ξ;
```

VI. Experimental evaluation

We evaluate the performance of the proposed algorithms and compare them with the state-of-the-art random walk based algorithms for graph sampling. All the algorithms are implemented in C++ and tested on a PC with Windows XP, 2xDual-Core Intel Xeon 2.66 GHz CPU and 4GB memory.

Datasets. We use four publicly available real-world datasets, which are Digg, WikiTalk, WebStanford, and RoadnetCA. (1) Digg is a social network dataset [29], (2) WikiTalk is a communication network dataset [30], (3) WebStanford is a web graph dataset [30], and (4) RoadnetCA is a road network dataset [30]. The detailed information of these datasets are reported in Table I. Note that the fourth column in Table I denotes the total variance distance (TVD) [11] between the stationary distribution of the random walk and the uniform distribution, i.e., $\text{TVD} = \frac{1}{2}\sum_{u \in V} |\frac{d_u}{2m} - \frac{1}{n}|$. As can be seen, the TVD of the first three datasets is large, implying a large deviation between the stationary distribution of the random walk and the uniform distribution. For the last dataset, the TVD is relatively small, as the node degrees in the road network are not very different from one another.

Evaluation methodology. Following most graph sampling literature [9], [1], [2], we evaluate different algorithms based on their estimation accuracies of degree distribution of a graph, because the degree distribution is a fundamental property of a graph. In particular, we use a commonly-used metric, the total variance distance (TVD), to measure the estimation accuracy of degree distribution. The TVD metric is a standard distance measure for two probability distributions. The larger TVD the algorithm achieves, the lower estimation accuracy it gets. For all the algorithms, we discard the first 5,000 samples (the so-called burn-in period) to ensure that the random walk converges to the stationary distribution. A similar process has been done in previous graph sampling studies [1], [2]. To guarantee the accuracy, we perform 1,000 independent simulations for all the algorithms. All the data points reported in the experiments are the average values over the 1,000 simulations. Below, we first compare our algorithms with the baselines given that parameters α (in RCMH and RCMHDA) and C (GMD and NGMD) are fixed to 0.1 and $0.5 \times d_{\max}$ respectively. Then, we present a comprehensive study on how such parameters affect the performance of our algorithms, and we also provide an empirically recommendation for setting such parameters in practice.

Exp-1: Comparison of various random walk samplers.

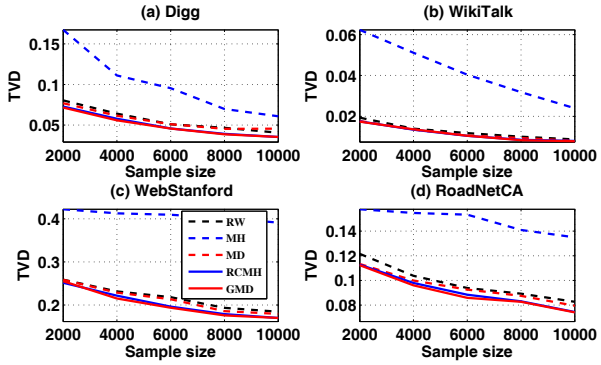

Fig. 3: Comparison of basic random walk samplers

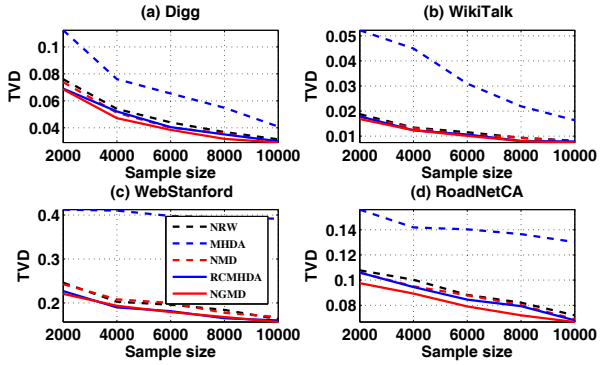

Fig. 4: Comparison of optimized random walk samplers

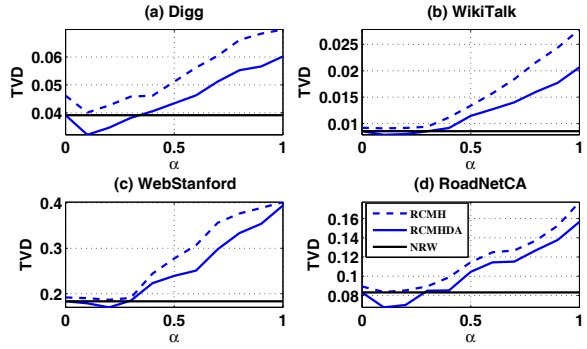

Fig. 5: Estimation accuracy vs α

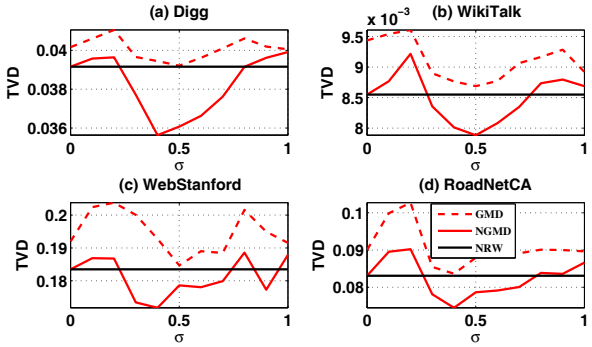

Fig. 6: Estimation accuracy vs $\sigma = C/d_{\max}$

First, we compare the estimation accuracy of different *basic* random walk samplers, which are the RW algorithm [1], the MH algorithm [10], [1], the MD algorithm [7], the proposed RCMH algorithm, and the proposed GMD algorithm. The results with varying sample size are reported in Fig. 3. As can be seen, the estimation accuracies of the GMD and RCMH algorithms are comparable, and they are consistently better than the other algorithms, which confirm the theoretical analysis presented in Section III and Section IV. As desired, the estimation accuracy of all the algorithms increase with increasing sample size. The MH algorithm performs poorly, which is consistent with the previous observations [4].

Second, we compare the performance of *optimized* random walk samplers, including the NRW algorithm [2], the MHDA algorithm [2], the proposed NMD algorithm, the proposed RCMHDA algorithm, and the proposed NGMD algorithm. Fig 4 depicts the results. We can see that the proposed RCMHDA and NGMD algorithms consistently outperform the other competitors over all the datasets. The estimation accuracy of NGMD is slightly better than that of RCMHDA in all datasets. These results imply that our algorithms needs a small number of samples to achieve the same estimation accuracy as the state-of-the-art NRW algorithm. For example, in WebStanford dataset (Fig 4(c)), the NRW algorithm requires 1.76 times more samples on average to achieve the same estimation accuracy as the NGMD algorithm. In addition, when comparing the optimized samplers with their basic counterparts, we can see that the optimized samplers consistently outperform the basic ones, which confirm the theoretical analysis in Section V.

Exp-2: Parameter effects and practical recommendation. In this experiment, we study how the parameters α (in RCMH and RCMHDA algorithms) and C (in GMD and NGMD algorithms) affect the performance of the proposed algorithms. Here we use the state-of-the-art NRW algorithm as the baseline, and set the sample size to 8,000. Similar results can also be observed with different sample sizes. For convenience, we set $\sigma = C/d_{\max}$, and use σ to control the parameter C in GMD and NGMD algorithms. Fig. 5 and Fig. 6 show the estimation accuracy of the proposed algorithms with varying α ($\in [0, 1]$) and σ ($\in [0, 1]$), respectively. From Fig. 5, we can see that in all datasets, the estimation accuracy of RCMH and RCMHDA generally increases as α increases when $\alpha \in [0, 0.2]$. When $\alpha \in [0.2, 1]$, the estimation accuracy of both RCMH and RCMHDA decrease with increasing α. Another finding is that if $\alpha \leq 0.3$, RCMHDA outperforms NRW. Hence, for practical recommendation, we suggest to set α to a value in the range $[0, 0.3]$ depending on different datasets. From Fig. 6, we observe that over all datasets, the estimation accuracy of NGMD is significantly better than that of NRW when $\sigma \in [0.3, 0.7]$, and when $\sigma \notin [0.3, 0.7]$ our algorithm is also comparable with NRW. As a result, for practical recommendation, we suggest to set σ in the range $[0.3, 0.7]$ depending on different datasets. Note that in many real-world graphs, $\sigma \in [0.3, 0.7]$ implies a large range of parameters C that can be selected. For example, in WebStanford datasets, $\sigma \in [0.3, 0.7]$ indicates $C \in [71, 165]$.

Besides the estimation accuracy, we also study how parameters α and C affect the sample acceptance ratios in the RCMH and RCMHDA algorithms as well as the sample repeated ratios in the GMD and NGMD algorithms, respectively. Fig. 7

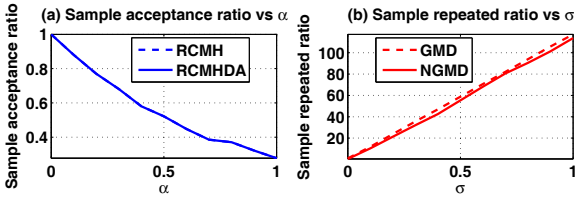

Fig. 7: Sample acceptance (repeated) ratio vs α (σ)

reports the results in WebStanford dataset with sample size 8,000. Similar results can also be observed in other datasets and with various sample size as well. From Fig. 7(a), we can clearly see that the acceptance ratio monotonically decreases as α increases. Note that the acceptance ratio of RCMH and RCMHDA are equivalent. When $\alpha = 1$, RCMH is downgraded to MH. This result confirms that the proposed RCMH and RCMHDA algorithms indeed reduce the rejection ratio of the MH algorithm. In addition, it is worth mentioning that the acceptance ratio of MH is only 0.278. This could be the major reason why MH performs poorly. From Fig. 7(b), we can observe that the sample repeated ratio increases with increasing σ (or C). Recall that when $\sigma = 1$, the GMD algorithm becomes the MD algorithm. Hence, compared to MD, our algorithms cuts the number of repeated samples given that $\sigma < 1$. These results validate the theoretical analysis presented in Section IV (Theorem 4.1). In addition, the sample repeated ratio of NGMD is slightly smaller than that of GMD which further confirms that the non-backtracking walk is better than the traditional random walk for unbiased graph sampling.

VII. Conclusion

In this paper, we first present a detailed analysis of the drawbacks of previous random walk based graph sampling algorithms. Then, we propose two novel algorithms, called rejection-controlled Metropolis-Hastings (RCMH) algorithm and generalized maximum-degree random walk (GMD) algorithm respectively. We show in theory and experiments that, both RCMH and GMD can balance the tradeoffs of the limitations of the previous algorithms by setting appropriate parameters. To further improve the estimation accuracy, we integrate the so-called delayed acceptance and non-backtracking random walk techniques into the RCMH and GMD algorithm, respectively. We conduct extensive experiments over four real-world graphs to evaluate the proposed algorithms, and the results show the effectiveness of the proposed algorithms.

Acknowledgment

The work was supported in part by (i) NSFC Grants (61402292, 61170076, U1301252, 61033009) and Natural Science Foundation of SZU (grant no. 201438); (ii) ARC DE140100999; (iii) Research Grants Council of the Hong Kong SAR, China, 14209314 and 418512; (iv) China 863 (no. 2012AA010239) and Guangdong Key Laboratory Project (2012A061400024).

References

[1] M. Gjoka, M. Kurant, C. T. Butts, and A. Markopoulou, "Practical recommendations on crawling online social networks," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 9, pp. 1872–1892, 2011.

[2] C.-H. Lee, X. Xu, and D. Y. Eun, "Beyond random walk and metropolis-hastings samplers: why you should not backtrack for unbiased graph sampling," in *SIGMETRICS*, 2012.

[3] J. Leskovec and C. Faloutsos, "Sampling from large graphs," in *KDD*, 2006.

[4] M. Kurant, A. Markopoulou, and P. Thiran, "Towards unbiased bfs sampling," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 9, pp. 1799–1809, 2011.

[5] Y.-Y. Ahn, S. Han, H. Kwak, S. B. Moon, and H. Jeong, "Analysis of topological characteristics of huge online social networking services," in *WWW*, 2007.

[6] A. S. Maiya and T. Y. Berger-Wolf, "Benefits of bias: towards better characterization of network sampling," in *KDD*, 2011.

[7] Z. Bar-Yossef, A. C. Berg, S. Chien, J. Fakcharoenphol, and D. Weitz, "Approximating aggregate queries about web pages via random walks," in *VLDB*, 2000.

[8] M. J. Salganik and D. D. Heckathorn, "Sampling and estimation in hidden populations using respondent-driven sampling," *Sociological Methodology*, vol. 34, pp. 193–239, 2004.

[9] A. H. Rasti, M. Torkjazi, R. Rejaie, N. G. Duffield, W. Willinger, and D. Stutzbach, "Respondent-driven sampling for characterizing unstructured overlays," in *INFOCOM*, 2009.

[10] D. Stutzbach, R. Rejaie, N. G. Duffield, S. Sen, and W. Willinger, "On unbiased sampling for unstructured peer-to-peer networks," *IEEE/ACM Trans. Netw.*, vol. 17, no. 2, pp. 377–390, 2009.

[11] Z. Bar-Yossef and M. Gurevich, "Random sampling from a search engine's index," *J. ACM*, vol. 55, no. 5, 2008.

[12] J. S. Liu, *Monte Carlo Strategies in Scientific Computing (Springer Series in Statistics)*. Springer, 2001.

[13] Z. Zhou, N. Zhang, Z. Gong, and G. Das, "Faster random walks by rewiring online social networks on-the-fly," in *ICDE*, 2013.

[14] S. J. Hardiman and L. Katzir, "Estimating clustering coefficients and size of social networks via random walk," in *WWW*, 2013.

[15] L. Katzir, E. Liberty, and O. Somekh, "Estimating sizes of social networks via biased sampling," in *WWW*, 2011.

[16] A. Dasgupta, R. Kumar, and T. Sarlós, "On estimating the average degree," in *WWW*, 2014, pp. 795–806.

[17] L. Addario-Berry and T. Lei, "The mixing time of the newman: Watts small world," in *SODA*, 2012.

[18] A. Mislove, M. Marcon, P. K. Gummadi, P. Druschel, and B. Bhattacharjee, "Measurement and analysis of online social networks," in *Internet Measurement Comference*, 2007.

[19] A. S. Maiya and T. Y. Berger-Wolf, "Sampling community structure," in *WWW*, 2010.

[20] B. F. Ribeiro and D. F. Towsley, "Estimating and sampling graphs with multidimensional random walks," in *Internet Measurement Conference*, 2010.

[21] K. Avrachenkov, B. F. Ribeiro, and D. F. Towsley, "Improving random walk estimation accuracy with uniform restarts," in *WAW*, 2010.

[22] D. Aldous and J. Fill, *Reversible Markov Chains and Random Walk on Graphs*, Monograph in preparation.

[23] S. Goel and M. J. Salganik, "Respondent-driven sampling as markov chain monte carlo," *Statistics in Medicine*, vol. 28, pp. 2202–2229, 2009.

[24] Z. Bar-Yossef and M. Gurevich, "Efficient search engine measurements," *TWEB*, vol. 5, no. 4, p. 18, 2011.

[25] ——, "Mining search engine query logs via suggestion sampling," *PVLDB*, 2008.

[26] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *Journal of Chemical Physics*, vol. 21, no. 6, pp. 1087–1092, 1953.

[27] W. K. Hastings, "Monte carlo sampling methods using markov chains and their applications," *Biometrika*, vol. 57, no. 1, pp. 97–109, 1970.

[28] R. M. Neal, "Improving asymptotic variance of mcmc estimators: non-reversible chains are better," *Technical report, No. 0406, Department of Statistics, University of Toronto*, 2004.

[29] R. Zafarani and H. Liu, "Social computing data repository at ASU," 2009. [Online]. Available: http://socialcomputing.asu.edu

[30] J. Leskovec, "Standford network analysis project," 2010. [Online]. Available: http://snap.standford.edu