Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 6783949, 12 pages
https://doi.org/10.1155/2020/6783949

Research Article

Differential Privacy for Evolving Network Based on GHRG

Jing Yang, Yuye Wang, and Jianpei Zhang

College of Computer Science and Technology, Harbin Engineering University, Harbin, Heilongjiang, China

Correspondence should be addressed to Yuye Wang; wangyuye@hrbeu.edu.cn

Received 21 July 2020; Revised 9 October 2020; Accepted 29 October 2020; Published 3 December 2020

Academic Editor: Ruben Specogna

Releasing evolving networks which contain sensitive information could compromise individual privacy. In this paper, we study the problem of releasing evolving networks under differential privacy. We explore the possibility of designing a differentially private evolving networks releasing algorithm. We found that the majority of traditional methods provide a snapshot of the networks under differential privacy over a brief period of time. As the network structure only changes in local part, the amount of required noise entirely is large and it leads to an inefficient utility. To this end, we propose GHRG-DP, a novel differentially private evolving networks releasing algorithm which reduces the noise scale and achieves high data utility. In the GHRG-DP algorithm, we learn the online connection probabilities between vertices in the evolving networks by *generalized hierarchical random graph* (GHRG) model. To fit the dynamic environment, a dendrogram structure adjusting method in local areas is proposed to reduce the noise scale in the whole period of time. Moreover, to avoid the unhelpful outcome of the connection probabilities, a Bayesian noisy probabilities calculating method is proposed. Through formal privacy analysis, we show that the GHRG-DP algorithm is ε-differentially private. Experiments on real evolving network datasets illustrate that GHRG-DP algorithm can privately release evolving networks with high accuracy.

1. Introduction

As more and more individual information in social network is published, releasing network data might pose threats to individual's privacy. To protect the privacy of individual information, network data should be sanitized before releasing. Differential privacy has been proposed to address such problem. Unlike the anonymization-based private algorithms (e.g., k-anonymity [1] and l-diversity [2]), differentially private algorithms are randomized algorithms. Differentially private networks releasing algorithms protect the essential structural information and ensure that the connection relationship of each participant is insensitive.

In the differentially private static network literature, the standard technique is to add noise to the output of an otherwise non-private algorithm. However, small changes of the network structure should cause a lot of impact point to query answers and it leads to a large amount of perturbation noise. Thus, synthetic network generation methods are used to substitute these standard methods. Roughly speaking, such synthetic network generation methods involve two steps: (1) using the real network to learn the parameters for a model of the network; (2) using the learned model to draw synthetic network from its probability distribution. These synthetic network generation methods could "dilute" the impact of small changes of the network structure by capturing the connection probabilities between vertices.

However, the interactions between vertices are often dynamic in nature, and how the network changes deserves to be taken into account. Given a sequence of evolving networks, the standard technique of differential privacy is to add noise to each snapshot of the evolving networks. In this way, each snapshot of the evolving networks satisfies differential privacy individually and turns the static differential privacy algorithm into a "dynamic" version. However, this kind of method suffers from poor performance. The root cause is it leads to a large amount of repetitive perturbation noise. We observe, in practice, the network structure between sequent snapshot changes within a local area. The remaining part of the network is almost unchanged and adding repetitive noise to the same connection probabilities is inefficient. Furthermore, the connection probabilities

between vertices of the network may obtain unhelpful outcomes when the connection probabilities equal 0 or 1.

To this end, we propose a novel differentially private evolving networks releasing algorithm, called GHRG-DP (i.e., differentially private based on generalized hierarchical random graph). In particular, we first use *generalized hierarchical random graph* (GHRG) model to represent the structure of evolving networks and utilize a hierarchical structure (called *dendrogram*) of the GHRG model to record connection probabilities between any pair of vertices which have appeared over a period of time. Unlike the popular hierarchical random graph (HRG) model, GHRG allows tree nodes of dendrogram to have any number of children and quantify the change of network's structure. As the network structure between sequent snapshots changes within a local area, it is important to design a method to generate a good dendrogram of each snapshot of evolving networks. We do not directly generate the best-fitting GHRG by MCMC method, but we generate the GHRG by adjusting the GHRG of the prior time. Then, we model each connection probability as a distribution instead of a point value and add noise to the parameters for the model of the network instead of adding noise to the point value of the connection probability. Finally, we compute connection probabilities' posterior by Bayesian theory.

To make the GHRG-DP satisfy differential privacy, we first sample a GHRG by a Markov chain Monte Carlo (MCMC) method by exponential mechanism while satisfying differential privacy. Then, we adjust the GHRG by exponential mechanism while satisfying differential privacy as well. Thirdly, we add Laplace noise to the parameters for the model of the network. Through formal privacy analysis, we prove that GHRG-DP satisfies ε-differential privacy. In summary, we present several contributions:

(1) We introduce generalized hierarchical random graph model as an effective means of encoding the evolving networks and propose a method to adjust the GHRG model which is suitable for evolving networks.

(2) We develop a novel differentially private evolving networks releasing algorithm, GHRG-DP. To ensure that the releasing evolving networks under differential privacy do not incur excessive noise, we propose a method by adding noise to the parameters for the model of the network.

(3) Through privacy analysis and experiments on real evolving network datasets, we prove that GHRG-DP algorithm can privately release evolving networks with high accuracy.

The rest of our paper is organized as follows. Section 2 provides a literature review. Section 3 introduces necessary background on differential privacy and hierarchical random graph model. Section 4 identifies the weakness of applying differentially private static network releasing algorithm to evolving networks. In Section 5, we describe our GHRG-DP algorithm in detail. Section 6 gives the privacy analysis. Section 7 reports the experimental results. Section 8 concludes the paper.

2. Related Work

Many existing works about social network differential privacy focus on social network analysis. These methods output some network statistics under differential privacy such as degree distribution, subgraph number, and clustering coefficient. In particular, Dwork et al. [3] proposed a differentially private method to answer the queries by adding noise to outcome directly. Hay et al. [4] proposed a differentially private method in a postprocessing phase and computed the consistent input most likely to have produced the noisy output. Hay et al. [5] used this method to estimate the degree distribution. Nissim et al. [6] introduced the concept of smooth sensitivity to protect individuals' privacy by adding a small amount of random noise to the released statistics, such as triangle count of a network. Zhang et al. [7] analyzed the released statistics through a ladder function and reduced the sensitivity effectively. Cheng et al. [8] presented a two-phase differentially private frequent subgraph mining algorithm called DFG. In DFG, frequent subgraphs are privately identified in the first phase, and the noisy support of each identified frequent subgraph is calculated in the second phase. Ding et al. [9] published the triangle counts satisfying the node-differential privacy with the triangle count distribution and the cumulative distribution. Sun et al. [10] formulated a decentralized differential privacy scheme named DDP, which requires that each participant considers not only their own privacy but also that of their neighbors involved in their ELV. Wang et al. [11] studied the problem of differential privacy for weighted network through the probability model.

Differentially private social network releasing algorithm also draws attention. Sala et al. [12] introduced a differentially private graph model called Pygmalion. Pygmalion extracts a graph's detailed structure into *dK*-graph, introduces noise into the resulting dataset, and generates a synthetic graph. Lu and Miklau [13] proposed algorithms for privately estimating the parameters of exponential random graph models (ERGMs) of a network. Based on the idea of stochastic Kronecker graph model, Mir and Wright [14] used maximum likelihood estimation to privately estimate the parameters. Xiao et al. [15] proposed a differentially private network publishing method, HRG-MCMC, witch computes an estimator of a given graph in the hierarchical random graph (HRG) model in a differentially private manner and samples possible HRG structures in the model space via Markov chain Monte Carlo (MCMC) witch satisfies the exponential mechanism. Qin et al. [16] investigated techniques to ensure local differential privacy of individuals while collecting structural information and generating representative synthetic social graphs. Chen et al. [17] presented a method for publishing differentially private synthetic attributed graphs, which is able to preserve the community structure of the original graph without sacrificing the ability to capture global structural properties.

Many existing works also focus on ensuring dynamic network data privacy. Bhagat et al. [18] analyzed the privacy risks of publishing multiple instances of the same network independently and proposed methods to perform group-

based anonymization. Tai et al. [19] introduced the concept of $k2$-degree anonymity, which limits the probability of a vertex being re-identified to $1/k$. Krzysztof Juszczyszy measured the transitions during network evolution by link prediction. Medforth and Wang [20] proposed an anonymization method to solve "degree-tail" attack. Macwan and Patel [21] proposed an anonymization approach to preserve the user identity from all the published time-series datasets of a social network. Yue et al. [22] proposed an efficient and adaptive general graph anonymization framework for incremental data publication. They proposed an anonymization process restart issue (APRI) and designed an activated function to determine whether an anonymization process should be restarted at a certain time in order to solve the problem of high time complexity and large information loss in the incremental data publication. Zhiyuli et al. [23] attempted to model the hierarchical and dynamic features of social networks by designing a damping-based sampling algorithm corresponding to a local search-based incremental learning algorithm, which can easily be extended to large-scale scenarios.

Most existing differentially private graph models do not fit evolving network. In this work, we develop a differentially private network publishing method for evolving network.

3. Background

3.1. Hierarchical Random Graph. Suppose the input network dataset is a simple undirected graph $G = (V, E)$, where V is the set of vertices and $E \subseteq V \times V$ is the set of edges. Claustet et al. [24] proposed the *hierarchical random graph* (HRG) model. HRG uses a hierarchical structure and connection probabilities between any pair of vertices to denote a graph G. The hierarchical structure is represented by a *dendrogram* T. The dendrogram is a binary tree which has n leaf nodes corresponding to the n vertices. Each internal node r represents a connection probability $p_r = E_r/N_r$. In particular, $N_r = n_{Lr} \cdot n_{Rr}$, where n_{Lr} and n_{Rr} denote the numbers of leaves of left subtrees L_r and right subtrees R_r, respectively. E_r is the number of edges between the leaves of L_r and R_r.

Given a graph G, we use the likelihood of the HRG to measure how much it matches G. The likelihood can be calculated as follows:

$$L(T, \{p_r\}) = \prod_{r \in T} p_r^{E_r} (1 - p_r)^{N_r - E_r}, \tag{1}$$

where $h(p_r) = -p_r \log p_r - (1 - p_r)\log(1 - p_r)$ is the Gibbs–Shannon entropy function. A higher likelihood corresponds to a better representation of the network's structure.

3.2. Generalized Hierarchical Random Graph. Peel and Clauset [25] introduced the generalized hierarchical random graph (GHRG) model in 2014. It also defines a couple of data structures $(T, \{p_r\})$ in formal. Unlike the popular HRG model, the GHRG allows the nodes in the dendrogram to have two or more child nodes. It models community structure at all scales and provides accurate fits to social networks. In the dynamic evolving systems, the GHRG could improve the interpretability for varying a network's structure.

3.3. Differential Privacy. Given a graph G, differential privacy ensures that the outputs are approximately the same even if any edge is arbitrarily added or deleted in the graph. Thus, the presence or absence of any edge has a negligible effect on the outputs. We define two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ to be neighbors if they satisfy $V_1 = V_2$, $E_1 \subset E_2$, and $|E_2| = |E_1| + 1$. ε-differential privacy is defined as follows.

Definition 1. (ε-differential privacy [3]). A randomized algorithm $\mathbf{A}$ is ε-differential privacy if for any two neighboring graphs G_1 and G_2, and for any output $O \in \mathrm{Range}(\mathbf{A})$,

$$p_r[\mathbf{A}(G_1) \in O] \leq e^{\varepsilon} p_r[\mathbf{A}(G_2) \in O]. \tag{2}$$

Differential privacy is based on the concept of *global sensitivity* of a function f. It is used to measure the maximum change in the outputs of f when any edge in the graph is changed. The global sensitivity of f is defined as $\Delta f = \max_{G_1, G_2} \| f(G_1) - f(G_2) \|_1$.

Differential privacy can be achieved by Laplace mechanism and exponential mechanism. The Laplace mechanism is mainly used for functions whose outputs are real values. Differential privacy can be achieved by adding properly noise drawn randomly from Laplace distribution to the true answer.

Theorem 1. *(Laplace mechanism [3]). For any function $f: G \longrightarrow R^d$ with sensitivity Δf, the algorithm*

$$\mathbf{A}(G) = f(G) + \langle \mathrm{Lap}_1(\Delta f/\varepsilon), \ldots, \mathrm{Lap}_d(\Delta f/\varepsilon) \rangle, \tag{3}$$

satisfies ε-differential privacy, where $Lap_i(\Delta f/\varepsilon)$ are i.i.d. Laplace variables with scale parameter $\Delta f/\varepsilon$.

The exponential mechanism is mainly used for functions whose outputs are not real numbers. The main idea is to sample the output data O from the output space O according to the utility function u. The global sensitivity of u is $\Delta u = \max_{O, G_1, G_2} |u(G_1, O) - u(G_2, O)|$.

Theorem 2. *(exponential mechanism [26]). Given a graph G and a utility function u: $(G \times \mathrm{O}) \longrightarrow R$, the arithmetic $\mathbf{A}$ whose output is with probability proportional to $\exp(\varepsilon \cdot u(G, O)/2\Delta u)$ satisfies ε-differential privacy.*

Theorem 3. *(sequential composition [27]). If each arithmetic $\mathbf{A}_i$ provides ε_i-differential privacy, a sequence of $(\mathbf{A}_1(D), \mathbf{A}_2(D), \ldots, \mathbf{A}_3(D))$ over the same database D provides $\sum_{i=1}^{n} \varepsilon_i$-differential privacy.*

4. A Straightforward Approach

In this section, we present a straightforward approach to public evolving networks under differential privacy. The main idea of this straightforward approach is to add noise to

every snapshot of the evolving network and let the graph of every timestamp satisfy differential privacy. In particular, we first identify the hierarchical random graph (HRG) that best fits G_t. G_t presents the snapshot of evolving networks at time t. And then, we generate $\widetilde{G}_t$ from the identified HRG. Recall that an HRG consists of a dendrogram T and associated probabilities set $\{p_r\}$. According to the HRG-MCMC method in [15], we can sample a good dendrogram T by means of Markov chain Monte Carlo (MCMC) procedure which satisfies exponential mechanism. For the associated probabilities set, we add noise to them directly to generate $\{\widetilde{p}_r\}$. At last, we generate $\widetilde{G}_t$ from $\{\widetilde{p}_r\}$ and dendrogram T. To satisfy differential privacy, the acceptance ratio p_{accept} is

$$p_{\text{accept}} = \min\left(1, \frac{\exp\left(\varepsilon_1 \log L\left(T'\right)/2\Delta u\right)}{\exp\left(\varepsilon_1 \log L\left(T\right)/2\Delta u\right)}\right), \tag{4}$$

where Δu is the global sensitivity of the utility function. It is worth noting that the generating $\widetilde{G}_t$ process of timestamp are independent of each other.

4.1. Privacy Analysis of Straightforward Approach. Firstly, at every timestamp, MCMC procedure under exponential mechanism satisfies ε_1−differential privacy, and the sensitivity is $\Delta u = \log N_{\max} + \log\left(1 + 1/\left(N_{\max} - 1\right)\right)^{N_{\max}-1} = O(\log n)$. When n is even $N_{\max} = n^2/4$, and $N_{\max} = (n^2 - 1)/4$ when n is odd. Then, generating $\{\widetilde{p}_r\}$ satisfies ε_2−differential privacy. Hence, based on the sequential composition, the method in each timestamp satisfies $(\varepsilon_1 + \varepsilon_2)$−differential privacy. When the time span is t, the straightforward approach satisfies $t(\varepsilon_1 + \varepsilon_2)$−differential privacy.

4.2. The Disadvantage of Straightforward Approach. However, the straightforward approach produces poor results. Firstly, as we observe, the structure of evolving network changes a little most of the time. The structure and the dendrogram of a snapshot are similar to a contiguous snapshot. The straightforward approach samples dendrogram T via MCMC procedure in each timestamp, respectively. It results in a lot of repetitive computation and drastically reduces the utility of the results. Then, the associated probability p_r may lead to calculation impairments when we calculate the likelihood function. Consider the case where zero connections $E_r = 0$, or all connections $E_r = N_r$; we have $p_r = 0$ or 1; thus the likelihood drops to 0 and results in an unhelpful outcome.

5. GHRG-DP Algorithm in Description

We propose a differentially private evolving networks releasing method, called GHRG-DP. This method is based on generalized hierarchical random graph and can solve the deficiency discussed in Section 4. Our solution is composed of two schemes: (1) we propose a dendrogram construction method for evolving network, called dendrogram construction based on adjustment (DCBA) (see Section 5.1). DCBA samples a dendrogram T^t_{sample} by MCMC process at t time. To achieve differential privacy, we employ the exponential mechanism during the MCMC process. Then, we propose a dendrogram structure adjustment method. We structure a dendrogram T^{t+1}_{sample} of $t+1$ time by adjusting the structure of T^t_{sample} locally. (2) Given the graph G_t and the dendrogram T^t_{sample}, we propose a noisy probability calculation method based on Bayesian theory (see Section 5.2). In this method, p_r will not be a single numerical value, but a distribution. Thus, we could quantify the uncertainty of p_r and avoid p_r equal to 0 or 1. Then, we calculate the posteriori distribution of p_r by Bayesian theory. To achieve differential privacy, we add noise to the parameters or hyperparameters to achieve the noisy probability $\{\widetilde{p}_r\}$. At last, we place edges to $\widetilde{G}_t$ with probability $\widetilde{p}_r$ and get a privatized synthetic graph.

5.1. Dendrogram Construction Method: DCBA. Given evolving graphs $\{G\}^1_n$, we measure how plausible this dendrogram is to represent the graph G_t by the logarithm of the likelihood. This log-likelihood can be computed by Bayes theorem. In fact, we find that the structures of adjacent dendrogram are similar. For example, in dataset AS-733 [28], the log-likelihood of dendrograms in adjacent time changes less than 0.05. Thus, we could get T^{t+1}_{sample} through adjusting T^t_{sample} locally and reduce the amount of noise we need to add.

To this end, we propose a locally dendrogram structure adjustment algorithm, called DCBA (dendrogram construction based on adjustment). The main idea of DCBA is structuring the dendrogram by GHRG model to satisfy the dynamic. When the structure of G_{t+1} changes a little compared to G_t, we could get T^{t+1}_{sample} through adjusting T^t_{sample} locally. Otherwise, we should sample a new T^{t+1}_{sample} through the MCMC algorithm which satisfies the differential privacy. In DCBA, we adjust the structure of dendrogram locally in order to simplify process of structuring the dendrogram. In particular, when a new vertex s is added to G_{t+1}, based on the GHRG model, we could first, respectively, insert s into T^t_{sample} as the brother node of each leaf node and get a candidate set of dendrogram. For example, Figure 1 gives an example of the candidate set of dendrogram. Then, we compute the log-likelihood of each candidate. Finally, we choose the dendrogram which has the maximum log-likelihood value to be T^{t+1}_{sample}.

The DCBA algorithm is shown in Algorithm 1. Given the evolving graphs $\{G\}^1_n$, we first judge the change situation of the graph structure at time $t+1$ through change-point detection method (line 1). We use the method from [25] to detect the change point and choose threshold. Next, when the change of the graph structure is greater than the threshold, we should sample the dendrogram renewedly through MCMC which simulates the exponential mechanism (lines 2–10). Let the utility function be $u(T) = \log L(T)$, where Δu is the sensitivity of the utility function. On the contrary, when the change of the graph structure is less than the threshold, we should adjust the structure of dendrogram locally. We get a candidate set of dendrogram and compute the log-likelihood of each candidate (lines 12–15). Finally, we choose the dendrogram

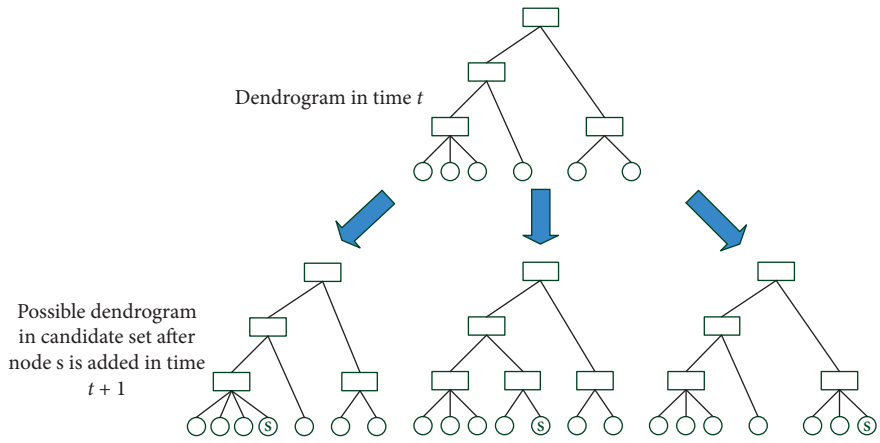

Figure 1: An example of the candidate set of dendrogram produced by GHRG.

Input : sampled dendrogram T^t_{sample} at time t, evolving graphs $\{G\}_n^1$, privacy parameter ε_1, ε_3.
Output : sampled dendrogram T^{t+1}_{sample} at time $t+1$.
(1) **if** the result of change-point detection of G_{t+1} is true **then**
(2) Initialize the Markov chain by choosing a random starting dendrogram T_0;
(3) **for** $i \in$ Markov chain step **do**
(4) Randomly pick an internal node r in T_{i-1};
(5) Pick a neighboring dendrogram T' of T_{i-1} by randomly drawing a configuration of r's subtrees;
(6) Accept the transition and set $T_i = T\prime$ with probability $\min(1, (\exp(\varepsilon_1 \log L(T')/2\Delta u_1)/\exp(\varepsilon_1 \log L(T_{i-1})/2\Delta u_1)))$;
(7) **end for**
(8) **if** equilibrium is reached **then**
(9) **Return** the sampled dendrogram $T^{t+1}_{\text{sample}} = T_i$;
(10) **end if**
(11) **else**
(12) **for each** new node $s \in \{v, v \in G_{t+1} \text{ and } v \notin G_t\}$ **do**
(13) **for each** internal node u whose child nodes are leaves in T^t_{sample} **do**
(14) regard s as u'child node and construct a candidate set of the new dendrogram;
(15) compute the log-likelihood of each candidate $\log L(T_{\text{candidate}})$;
(16) **end for**
(17) sample the new dendrogram with probability proportional to $\exp(\varepsilon_3 \cdot \log L(T_{\text{candidate}})/2)/\sum_{T'_{\text{candidate}} \in \mathbb{T}} \exp(\varepsilon_3 \cdot \log L(T'_{\text{candidate}})/2)$ from the candidate set;
(18) **end for**
(19) **Return** the sampled dendrogram T^{t+1}_{sample};
(20) **end if**

Algorithm 1: Dendrogram construction based on adjustment (DCBA).

through the exponential mechanism (line 17). Let the utility function be $u'(T_{\text{candidate}}) = \log L(T_{\text{candidate}})$ and we sample $T_{\text{candidate}}$ with probability proportional to $\exp(\varepsilon_3 \cdot \log L(T_{\text{candidate}})/\ 2\Delta u')/\sum_{T_{\text{candidate}}' \in \mathbb{T}} \exp(\varepsilon_3 \cdot \log L(T_{\text{candidate}}')/2\Delta u')$. $\mathbb{T}$ is the entire candidate set of the new dendrogram. It is easy to know that the global sensitivity $\Delta u' < 1$ and we set it as 1.

5.2. Noisy Probability Calculation Method: PCBD. The noisy connection probability computing algorithm PCBD is shown in Algorithm 2. To avoid connection probability p_r to be 0 or 1, we model p_r as a distribution and prevent its expected value from becoming 0 or 1. When the Beta distribution has hyperparameters as $\alpha = \beta = 1$, it corresponds to a uniform distribution. So, we treat the uniform distribution with the parameter p_r as the Beta distribution with hyperparameters $\alpha = \beta = 1$. In addition, the Beta distribution is conjugate with the Binomial distribution. When the conjugate prior distribution is the Beta distribution, the posteriori distribution is still the Beta distribution. And when the conjugate prior distribution is the uniform distribution with the parameter p_r, the posteriori distribution is the Beta distribution as well. We could get the likelihood through Bayesian theory:

$$p(G|T,\{p_r\}) = p(G|T,\alpha,\beta) = \prod_r \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \frac{\Gamma(E_r+\alpha)\Gamma(N_r-E_r+\beta)}{\Gamma(N_r+\alpha+\beta)}. \tag{5}$$

Given the evolving graphs in the sliding window $\{G_t,\ldots,G_{t+w-1}\}$, where w is the length of the sliding window, we could compute the posteriori distribution $p_{r\mathrm{MAP}}$ by updating the hyperparameters. $p_{r\mathrm{MAP}}$ represents the maximum value of p_r using Maximum A Posteriori (MAP). We update the hyperparameters as follows:

$$\begin{aligned}\alpha_r' &= \alpha + \sum_{\{G_t,\ldots,G_{t+w-1}\}} E_r^{G_t},\\ \beta_r' &= \beta + \sum_{\{G_t,\ldots,G_{t+w-1}\}} N_r - E_r^{G_t}.\end{aligned} \tag{6}$$

Thus, we could obtain $p_{r\mathrm{MAP}}$ from $(E_r+\alpha_r')/(N_r+\alpha_r'+\beta_r')$. To satisfy the differential privacy, we add noise to the hyperparameters α_r' and β_r' with application of the Laplace mechanism. The global sensitivity of PCBD is $2N$, where N is the size of the node set which contains the nodes of all the graphs in the sliding window. The reason is that when we insert or delete any edge, the maximum effect it leads to is N. The sanitized hyperparameters are $\widetilde{\alpha}_r' = \alpha_r' + \mathrm{lap}(2N/\varepsilon_2)$ and $\widetilde{\beta}_r' = \beta_r' + \mathrm{lap}(2N/\varepsilon_2)$, and

$$\widetilde{p}_{r\mathrm{MAP}} = \frac{E_r+\widetilde{\alpha}_r'}{N_r+\widetilde{\alpha}_r'+\widetilde{\beta}_r'}. \tag{7}$$

The PCBD algorithm is shown in Algorithm 2. In particular, we first get the union set G_t' which contains all the graphs in the sliding window and compute N_r from G_t'. We should also compute E_r from G_t (lines 1-2). Then, we compute the increment of the hyperparameters $\Delta\alpha_i$ and $\Delta\beta_i$ using N_r and E_r of each graph in the sliding window (lines 3-4). To satisfy the differential privacy, we add noise to $\Delta\alpha_i$ and $\Delta\beta_i$, and we get $\Delta\widetilde{\alpha}_i$ and $\Delta\widetilde{\beta}_i$ (line 5). Next, we update the hyperparameters $\widetilde{\alpha}_r'$ and $\widetilde{\beta}'$ and compute $\widetilde{p}_{r\mathrm{MAP}}$(lines 7-8). Finally, we use recursion method to compute the noisy probabilities of internal nodes of all the dendrograms.

5.3. Generation of Synthetic Graph. After getting the sampled dendrogram T^t_{sample} and the noisy probabilities $\{\widetilde{p}_{r\mathrm{MAP}}\}_t$, for each pair of nodes $i, j \in V_t$, we could find the common ancestor r of i, j which is closest to them. Then, we place an edge E_{ij} with independent probability $\widetilde{p}_{r\mathrm{MAP}}$ and get the synthetic graph $\widetilde{G}_t$. Algorithm 3 shows the specific process.

6. Privacy Analysis of GHRG-DP

6.1. Sensitivity Computation of GHRG-DP. We first analyze the global sensitivity Δu of DCBA algorithm. Δu is the maximum change in the log-likelihood when any edge of graph misses. From [15], we know $\Delta u = \max_{r\in T}|(-N_r h(p_r) - (-N_r h(p_r')))|$, where $p_r = (E_r + \alpha_r')/(N_r + \alpha_r' + \beta_r'), p_r' = (E_r - 1 + \alpha_r')/(N_r + \alpha_r' + \beta_r')$.

Theorem 4. *$\Delta u = \log((w+1)N_r - w + 1) + (1+w)\log(1+w) - (2+w)\log(2+w)$ and it is $O(\log n)$.*

Proof. We first fix N_r and treat E_r as variable. Let $f(E_r) = h(p_r) - h(p_r')$, and we have $\Delta u = \max|f(E_r)|$. For all p_r, we have $h''(p_r) < 0$. Then, we could know $h'(p_r) = \log(1-p_r) - \log p_r$ monotonically decreases. Since $h'(0.5) = 0$, we note that $h'(p_r) > 0$ when p_r is in (0,0.5), and $h'(p_r) < 0$ when p_r is in (0.5,1]. As $h(p_r)$ has symmetric property, we have $\Delta u = \max(-\min(N_r \cdot f(E_r)), \max(N_r \cdot f(E_r))) = \max(f(E_r))$. When $p_r = 1$ or 0, that is, $E_r = N_r$ or 1, $f(E_r)$ gets the maximum value. Let $E_r = 1$, and we have

$$\begin{aligned}\Delta u &= N_r\left|h(p_r) - h(p_r')\right|\\ &= N_r\left|p_r\log p_r + (1-p_r)\log(1-p_r) - (p_r'\log p_r' + (1-p_r')\log(1-p_r'))\right|\\ &= N_r\Bigg|\frac{E_r+\alpha_r'}{N_r+\alpha_r'+\beta_r'}\log\left(\frac{E_r+\alpha_r'}{N_r+\alpha_r'+\beta_r'}\right) + \left(1-\frac{E_r+\alpha_r'}{N_r+\alpha_r'+\beta_r'}\right)\log\left(1-\left(\frac{E_r+\alpha_r'}{N_r+\alpha_r'+\beta_r'}\right)\right)\\ &\quad - \frac{E_r-1+\alpha_r'}{N_r+\alpha_r'+\beta_r'}\log\frac{E_r-1+\alpha_r'}{N_r+\alpha_r'+\beta_r'} - \left(1-\frac{E_r-1+\alpha_r'}{N_r+\alpha_r'+\beta_r'}\right)\log\left(1-\left(\frac{E_r-1+\alpha_r'}{N_r+\alpha_r'+\beta_r'}\right)\right)\Bigg|\\ &= \frac{N_r}{N_r+\alpha_r'+\beta_r'}\left|(1+\alpha_r')\log(1+\alpha_r') + (N_r-1+\beta_r')\log(N_r-1+\beta_r') - \alpha_r'\log\alpha_r' - (N_r+\beta_r')\log(N_r+\beta_r')\right|\\ &= \frac{1}{w+1}\left(((w+1)N_r - w + 1)\log((w+1)N_r - w + 1) - ((w+1)N_r - w)\log((w+1)N_r - w) + k\right),\end{aligned} \tag{8}$$

Input: evolving graphs $\{G\}_{t+w-1}^{t}$, sampled dendrogram T_{sample}^{t}, privacy parameter ε_2, the length of the sliding window w, T_{sample}^{t}'s internal node r^*
Output: noisy probabilities $\{\widetilde{p}_{r\text{MAP}}\}_t$, $r \in \{r^*, \text{all internal nodes below } r^*\}$
(1) $G_t' = G_t \cup G_{t+1} \cup \cdots \cup G_{t+w-1}$;
(2) compute the N_r of G_r' and E_r of G_t;
(3) **for** $G_i \in \{G_t, G_{t+1}, \ldots, G_{t+w-1}\}$ do
(4) $\Delta\alpha_i = E_r^{G_i}, \Delta\beta_i = N_r - E_r^{G_i}$;
(5) $\Delta\widetilde{\alpha}_i = E_r^{G_i} + \text{lap}(2N/\varepsilon_2)$, $\Delta\widetilde{\beta}_i = N_r - E_r^{G_i} + \text{lap}(2N/\varepsilon_2)$;
(6) **for end**
(7) $\widetilde{\alpha}_r' = \alpha + \sum_{i=t,t+1,\ldots,t+w-1} \Delta\widetilde{\alpha}_i, \widetilde{\beta}_r' = \beta + \sum_{i=t,t+1,\ldots,t+w-1} \Delta\widetilde{\beta}_i$;
(8) $\widetilde{p}_{r\text{MAP}} = (E_r + \widetilde{\alpha}_r'/N_r + \widetilde{\alpha}_r' + \widetilde{\beta}_r')$;
(9) **for each** r^*'child **do**
(10) $r_c \leftarrow r^*$'child;
(11) PCBD $(\{G\}_{t+w-1}^{t}, T_{\text{sample}}^{t}, \varepsilon_2, w, r_c)$;
(12) **end for**

Algorithm 2: Probability calculation based on distribution (PCBD).

Input: input graph $\{G\}_{t+w-1}^{t}$, sampled dendrogram T_{sample}^{t}, privacy parameter ε_2
Output: synthetic graph $\widetilde{G}_t$
(1) $r_{\text{root}} \leftarrow$ root node of T_{sample}^{t} ;
(2) PCBD $(\{G\}_{t+w-1}^{t}, T_{\text{sample}}^{t}, \varepsilon_2, w, r_{\text{root}})$;
(3) **for** each pair of vertices $i, j \in V_t$ **do**
(4) find the lowest common ancestor r of i and j in T_{sample}^{t};
(5) place an edge in $\widetilde{G}_t$ between i and j with independent probability $\widetilde{p}_{r\text{MAP}}$;
(6) **for end**
(7) **return** $\widetilde{G}_t$;

Algorithm 3: Generate synthetic graph.$\widetilde{G}_t$.

where the constant $k = (1+w)\log(1+w) - (2+w)\log(2+w)$. Let $f(x) = x\log x$, we have $f'(x) = 1 + \log x$, and $\Delta u \leq \log((w+1)N_r - w + 1) + k + 1 = O(\log N_r) = O(\log n)$. □

6.2. Privacy Proof of GHRG-DP. Based on the sequential composition property, we prove that GHRG-DP satisfies ε-differential privacy.

Theorem 5. *GHRG-DP satisfies ε-differential privacy.*

Proof. We suppose that the length of evolving graphs is T, and there are m change points. Based on the sequential composition property, DCBA algorithm satisfies $(m\varepsilon_1 + (T-m)\varepsilon_3)$-differential privacy and PCBD algorithm satisfies $T\varepsilon_2$-differential privacy. In Algorithm 3, we know that the synthetic graph is generated through sampled dendrogram and noisy probabilities, and it does not consume any privacy budget. Hence, GHRG-DP satisfies ε-differential privacy, and $\varepsilon = (m\varepsilon_1 + (T-m)\varepsilon_3) + T\varepsilon_2$. □

7. Experimental Results

7.1. Experiment Settings. We make use of two real-world evolving network datasets in our experiments: AS-caida [28] and AS-733. The dataset contains 122 CAIDA AS graphs, from January 2004 to November 2007. Each file contains a full AS graph derived from a set of RouteViews BGP table snapshots. In our experiment, we extracted snapshots weekly from Jan. 2004 to Apr. 2006. AS-733 collects the communication record of an Autonomous System (AS). It is collected from University of Oregon RouteViews Project. In our experiment, we extracted snapshots daily from Nov. 8, 1997, to Dec. 17, 1997. The statistics of portion data are shown in Tables 1 and 2.

We compare the accuracy of GHRG-DP with DDPA [29] and the straightforward method through (1) the relative error of degree distribution and shortest path length distribution (2) the running time. The relative error is represented as

$$\text{rel.err} = \frac{\left|S(G) - S_{\text{avg}}(\widetilde{G})\right|_1}{2}. \tag{9}$$

As differential privacy needs to produce random noise, we measure the accuracy of the result by the median relative error where we run the Laplace mechanism 10 times.

7.2. Evaluation of GHRG-DP

7.2.1. Comparing with Other Approaches. To show the utility of GHRG-DP algorithm, we first compare the relative error of degree distribution and shortest path length distribution

Tabel 1: Statistics of AS-caida.

Snapshot	1	5	9	13	17	21	25	29	33	37	40
#Nodes	16301	17160	17848	18740	19489	20344	21202	21339	21548	21672	21885
#Edges	65910	70026	74344	77002	79718	81882	85850	86566	87784	88654	88944

Tabel 2: Statistics of AS-733.

Snapshot	1	5	9	13	17	21	25	29	33	37	40
#Nodes	3015	3022	3037	3055	3066	3095	3109	3130	3153	3166	3172
#Edges	5347	5368	5361	5433	5440	5533	5550	5774	5798	5869	5855

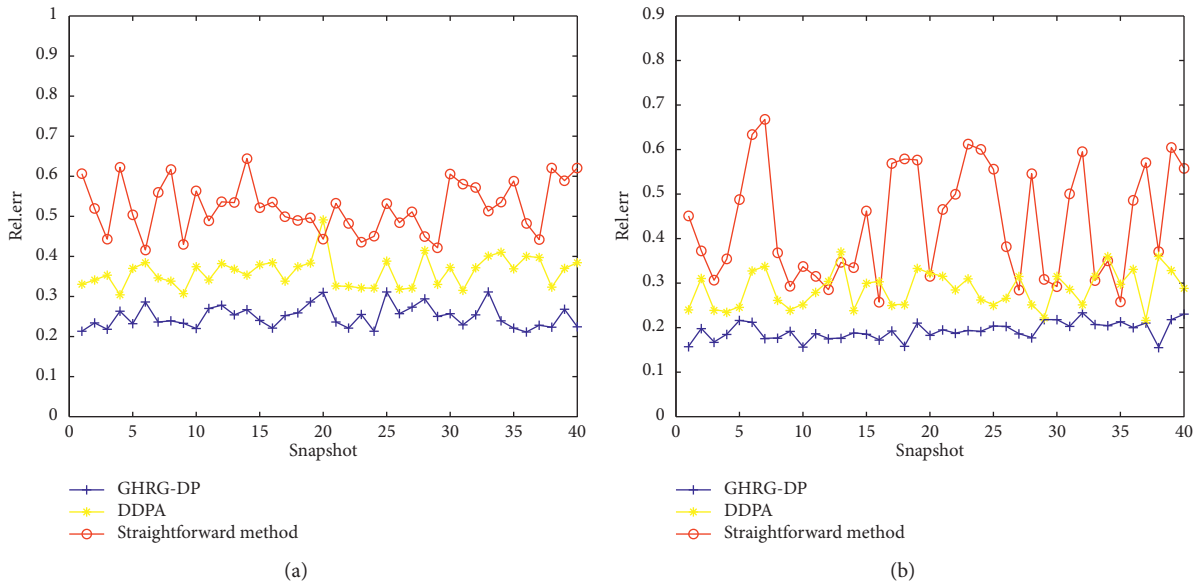

Figure 2: The relative error of degree distribution. (a) AS-caida. (b) AS-733.

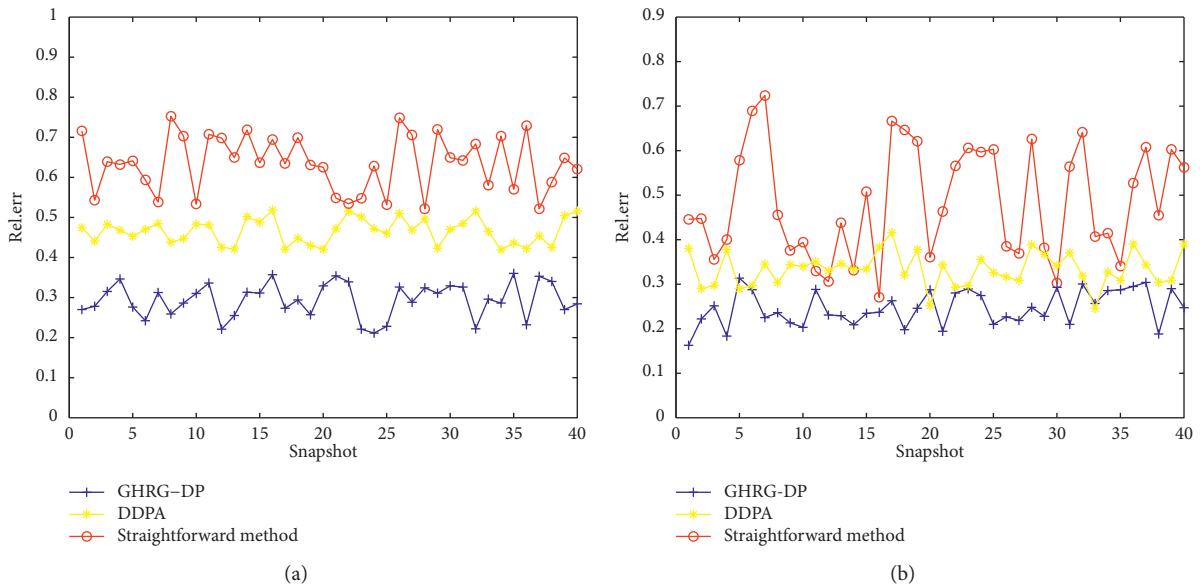

Figure 3: The relative error of shortest path length distribution. (a) AS-caida. (b) AS-733.

Tabel 3: Running time evaluation.

Dataset	Average time of GHRG-DP (ms)	Average time of DDPA (ms)	Average time of straightforward method (ms)
AS-caida	506	6144	5904
AS-733	88.5	1106	1054
Dataset	Median of GHRG-DP (ms)	Median of DDPA (ms)	Median of straightforward method (ms)
AS-caida	62	5933	5912
AS-733	10	1003	995

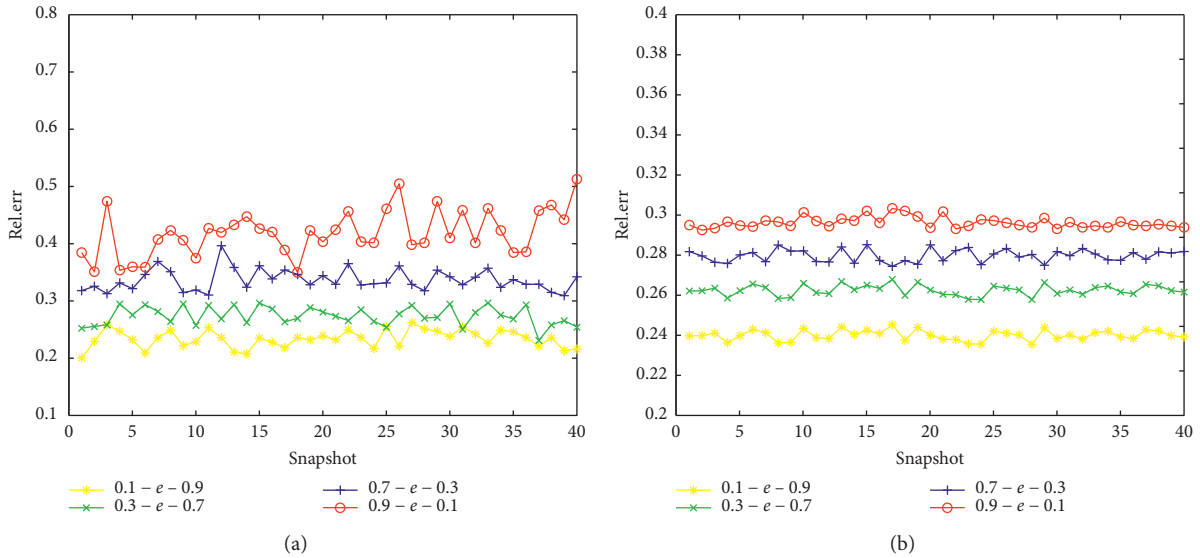

Figure 4: The relative error of degree distribution on different ε. (a) AS-caida. (b) AS-733.

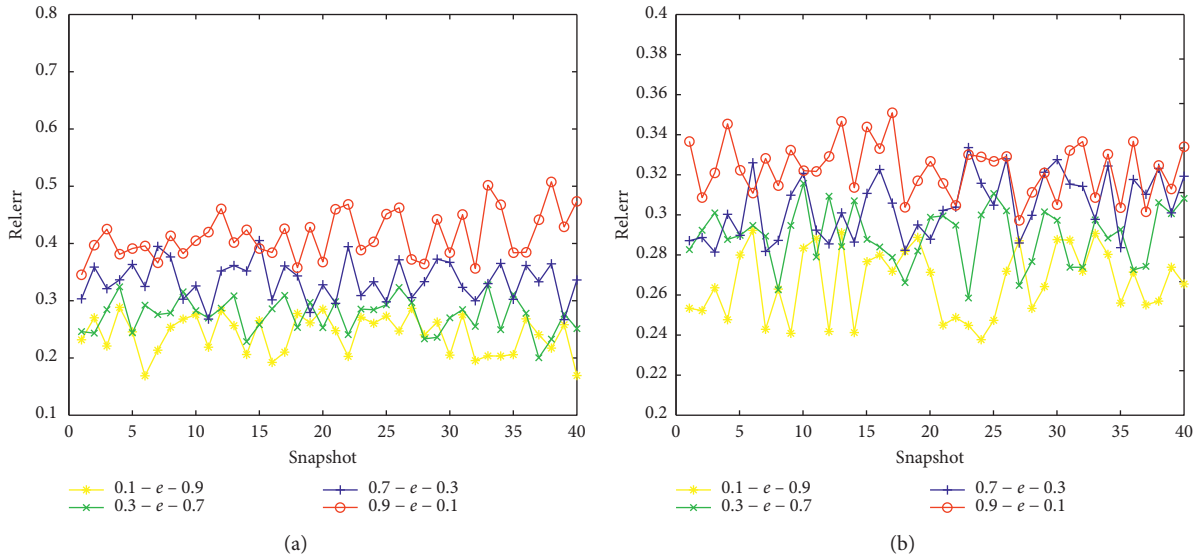

Figure 5: The relative error of shortest path length distribution on different ε. (a) AS-caida. (b) AS-733.

with DDPA and the straightforward method which we have mentioned above. From Figures 2 and 3, we can see that GHRG-DP outperforms the DDPA and straightforward method. This is because we add noise to the posterior parameters of the model instead of adding noise to the connection probabilities. GHRG-DP avoids perturbing the connection probabilities directly. Furthermore, we observe that, comparing the variance of the relative error during the entire time series, GHRG-DP also outperforms the straightforward method. This means that the output of GHRG-DP is more stable. This is because, during the process of PCBD algorithm, neighbor networks are similar to each other and it leads to the similar posterior parameters. Thus, the networks GHRG-DP generates are more stable.

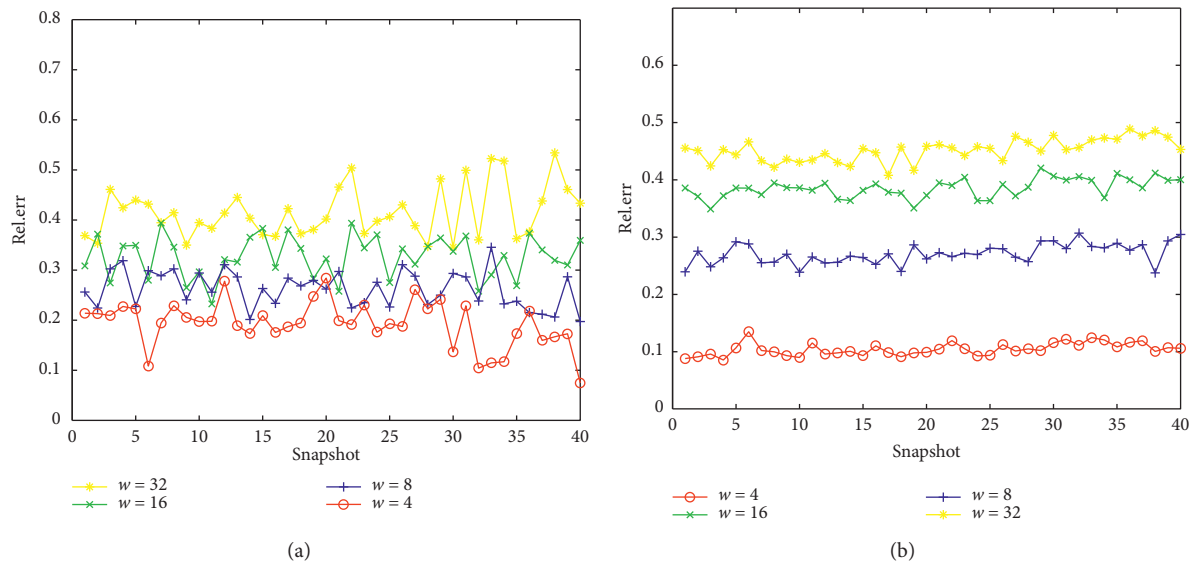

Figure 6: The relative error of degree distribution on different w. (a) AS-caida. (b) AS-733.

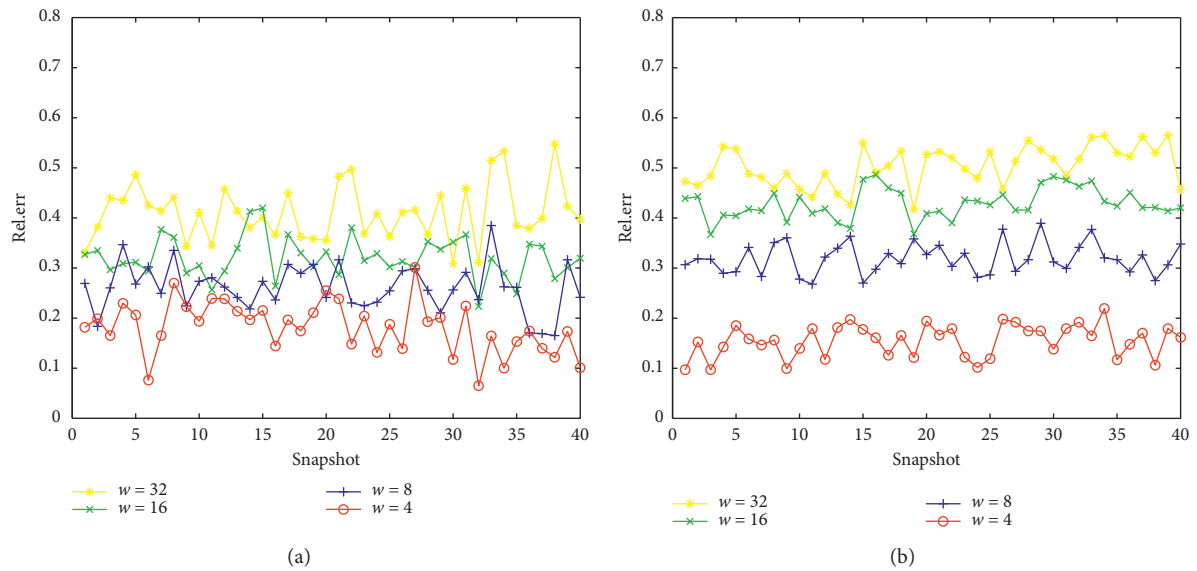

Figure 7: The relative error of shortest path length distribution on different w. (a) AS-caida. (b) AS-733.

Then, we compare the running time with DDPA and the straightforward method. From Table 3, we can see that GHRG-DP uses less time than DDPA and the straightforward method takes the most time. This is because, in GHRG-DP, we generate the dendrogram by adjustment in most time and it omits the MCMC process. However, the median of GHRG-DP is much less than the average, and the median of DDPA and the straightforward method is roughly equal to the average. This is because, when the result of change-point detection is false, the adjustment process of dendrogram omits the MCMC computing process. The running time is much less than the straightforward method. However, when the result of change-point detection is true, GHRG-DP needs to sample the dendrogram by MCMC. At this time, the running time is similar. As a result, the running time of GHRG-DP increases, and some part of running time becomes protruding.

7.2.2. Evaluation of GHRG-DP on Different ε. In Figures 4 and 5, we show how the assignment of privacy parameter affects the performance of GHRG-DP. We fix $\varepsilon = 1.0$ and assign $(\varepsilon_1, \varepsilon_2) = \{(0.1, 0.9), (0.3, 0.7), (0.6, 0.4), (0.9, 0.1)\}$ for DCBA and PCBD, respectively. We compare the relative error of degree distribution and shortest path length distribution. From Figures 4 and 5, we can see, when ε_1 is relatively large, the relative error always stays high. This is because, in PCBD, we perturb the model parameters and it affects the synthetic graph directly. This means that, compared with the process of sampling dendrogram under

differential privacy, the process of perturbing the model parameters affects the performance of GHRG-DP much more.

7.2.3. Evaluation of GHRG-DP on Different Lengths of Sliding Window. In Figures 6 and 7, we show how the length of the sliding window affects the performance of GHRG-DP. We set the length to be $w = \{4, 8, 16, 32\}$, respectively. We compare the relative error of degree distribution and shortest path length distribution.

Form Figures 6 and 7, we can see, when the length of the sliding window is relatively long, the relative error always stays high. This is because, when we update the hyperparameters in PCBD, we need to compute the summation of $\Delta\widetilde{\alpha}$ and $\Delta\widetilde{\beta}$ of all the nodes in the sliding window. It generates much cumulate noise.

8. Conclusions

In this paper, we investigate the problem of evolving network differential privacy. We first introduce a straightforward approach for publishing differentially private evolving network. This approach sanitizes each timestamp network, respectively. We observe that the amount of noise required in straightforward approach is really high. Thus, we propose our GHRG-DP algorithm. The main idea is adjusting the structure of dendrogram to reduce the amount of noise during the process of MCMC. We change the real value of connection to the distribution of connection probability. It satisfies the differential privacy through adding noise during the iteration process of updating the hyperparameters. It could fit the dynamic environment of evolving network. Formal privacy analysis and the results of extensive experiments show GHRG-DP can achieve a private evolving network with high utility. Our future research directions are the more effective change-point detection method in Algorithm 1 under differential privacy.

Data Availability

The data used to support the findings of this study can be accessed from http://snap.stanford.edu/.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under grant numbers 61672179, 61370083, and 61402126, in part by the Natural Science Foundation of Heilongjiang Province under grant number F2015030, in part by the Science Foundation for Youths of Heilongjiang under grant number QC2016083, and in part by the Postdoctoral Foundation of Heilongjiang Province under grant number LBH-Z14071.

References

[1] L. Sweeney, "k-anonymity: a model for protecting privacy," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 5, pp. 557–570, 2002.

[2] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam, "L-diversity: privacy beyond k-anonymity," in *Proceedings of the 22nd International Conference on Data Engineering (ICDE'06)*, p. 24, Atlanta, GA, USA, April 2006.

[3] C. Dwork, F. Mcsherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Proceedings of the 3rd Theory of Cryptography Conference*, pp. 265–284, New York, NY, USA, March 2006.

[4] M. Hay, V. Rastogi, and G. Miklau, "Boosting the accuracy of differentially private histograms through consistency," *Proceedings of VLDB Endowment*, vol. 3, no. 1-2, pp. 1021–1032, 2010.

[5] M. Hay, C. Li, G. Miklau, and D. Jensen, "Accurate estimation of the degree distribution of private networks," in *Proceedings of the 9th IEEE International Conference on Data Mining*, pp. 169–178, Miami, FL, USA, December 2009.

[6] K. Nissim, S. Raskhodnikova, and A. Smith, "Smooth sensitivity and sampling in private data analysis," in *Proceedings of the 39th Annual ACM Symposium on Theory of Computing*, pp. 75–84, SanDiego, CA, USA, June 2007.

[7] J. Zhang, G. Cormode, C. M. Procopiuc, D. Srivastava, and X. Xiao, "Private release of graph statistics using ladder functions," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 731–745, Melbourne, Australia, June 2015.

[8] X. Cheng, S. Su, S. Xu, L. Xiong, K. Xiao, and M. Zhao, "A two-phase Algorithm for differentially private frequent subgraph mining," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 8, pp. 1411–1425, 2018.

[9] X. Ding, X. Zhang, Z. Bao, and H. Jin, "Privacy-preserving triangle counting in large graphs," in *Proceedings of the Conference on Information and Knowledge Management*, pp. 1283–1292, Torino, Italy, October 2018.

[10] H. Sun, X. Xiao, I. Khalil et al., "Analyzing subgraph statistics from extended local views with decentralized differential privacy," in *Proceedings of the 26th ACM conference on Computer and Communications Security*, pp. 703–717, London, UK, November 2019.

[11] Y. Wang, J. Yang, and J. Zhang, "Differential privacy for weighted network based on probability model," *IEEE Access*, vol. 8, pp. 80792–80800, 2020.

[12] A. Sala, X. Zhao, C. Wilson, X. Zheng, and B. Y. Zhao, "Sharing graphs using differentially private graph models," in *Proceedings of the ACM SIGCOMM Conference on Internet Measurement Conference*, pp. 81–98, Berlin, Germany, November 2011.

[13] W. Lu and G. Miklau, "Exponential random graph estimation under differential privacy," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 921–930, New York, NY, USA, 2014.

[14] D. Mir and R. N. Wright, "A differentially private estimator for the stochastic kronecker graph model," in *Proceedings of the 2012 Joint EDBT/ICDT Workshops*, pp. 167–176, Berlin, Germany, March 2012.

[15] Q. Xiao, R. Chen, and K. Tan, "Differentially private network data release via structural inference," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge*

Discovery and Data Mining, pp. 911–920, New York, NY, USA, August 2014.

[16] Z. Qin, T. Yu, Y. Yang, I. Khalil, Y. Xiao, and K. Ren, "Generating synthetic decentralized social graphs with local differential privacy," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 425–438, Dallas, TX, USA, October 2017.

[17] X. Chen, S. Mauw, and Y. Ramírez-Cruz, "Publishing community-preserving attributed social graphs with a differential privacy guarantee," *Proceedings on Privacy Enhancing Technologies*, vol. 2020, no. 4, 2020.

[18] S. Bhagat, G. Cormode, B. Krishnamurthy, and D. Srivastava, "Privacy in dynamic social networks," in *Proceedings of the 19th International Conference on World Wide Web, 2010*, pp. 1059-1060, Raleigh, NC, USA, April 2010.

[19] C. H. Tai, P. S. Yu, D.-N. Yang, and M.-S. Chen, "Privacy-preserving social network publication against friendship attacks," in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1262–1270, San Diego, CA, USA, August 2011.

[20] N. Medforth and K. Wang, "Privacy risk in graph stream publishing for social network data," in *Proceedings of the IEEE 11th International Conference on Data Mining*, pp. 437–446, Vancouver, Canada, December 2011.

[21] K. R. Macwan and S. J. Patel, "Privacy preserving approach in dynamic social network data publishing," in *Proceedings of the Information Security Practice and Experience 15th International Conference, ISPEC 2019*, pp. 381–398, Kuala Lumpur, Malaysia, November 2019.

[22] R. Yue, Y. Li, T. Wang et al., "An efficient adaptive graph anonymization framework for incremental data publication," in *Proceedings of the 2018 5th International Conference on Behavioral Economic and Socio Cultural Computing (BESC 2018)*, pp. 103–108, Kaohsiung, Taiwan, November 2018.

[23] A. Zhiyuli, X. Liang, Y. Chen, and X. Du, "Modeling large-scale dynamic social networks via node embeddings," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 10, pp. 1994–2007, 2019.

[24] A. Clauset, C. Moore, and M. E. J. Newman, "Hierarchical structure and the prediction of missing links in networks," *Nature*, vol. 453, no. 7191, pp. 98–101, 2008.

[25] L. Peel and A. Clauset, "Detecting change points in the large-scale structure of evolving networks," in *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, pp. 2914–2920, Austin, TX, USA, January 2015.

[26] F. McSherry and K. Talwar, "Mechanism design via differential privacy," in *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*, pp. 94–103, Providence, RI, USA, October 2007.

[27] F. Mcsherry, "Privacy integrated queries," *Communications of The ACM*, vol. 53, no. 9, pp. 89–97, 2010.

[28] J. Leskovec and A. Krevl, "SNAP datasets: Stanford large network dataset collection," 2014, http://snap.stanford.edu/.

[29] Z. Liu, Y. Dong, X. Zhao, and B. Zhang, "A dynamic social network data publishing algorithm based on differential privacy," *Journal of Information Security*, vol. 8, no. 4, pp. 328–338, 2017.