Computer Communications 73 (2016) 176–187

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

More than topology: Joint topology and attribute sampling and generation of social network graphs

Michael Seufert*, Stanislav Lange, Tobias Hoßfeld[1]

Institute of Computer Science, University of Würzburg, Würzburg, Germany

ARTICLE INFO

Article history:
Available online 30 July 2015

Keywords:
Social networks
Attributes
Graph sampling
Graph similarity
Graph generation

ABSTRACT

Graph sampling refers to the process of deriving a small subset of nodes from a possibly huge graph in order to estimate properties of the whole graph from examining the sample. Whereas topological properties can already be obtained accurately by sampling, current approaches do not take possibly hidden dependencies between node topology and attributes into account. Especially in the context of online social networks, node attributes are of importance as they correspond to properties of the social network's users. Therefore, existing sampling algorithms can be extended to attribute sampling, but still lack the capturing of structural properties. Analyzing topology (e.g., node degree and clustering coefficient) and attribute properties (e.g., age and location) jointly can provide valuable insights into the social network and allows for a better understanding of social processes. As major contribution, this work proposes a novel sampling algorithm which provides unbiased and reliable estimates of joint topological and attribute based graph properties in a resource efficient fashion. Furthermore, the obtained samples allow for the generation of synthetic graphs, which show high similarity to the original graph with respect to topology and attributes. The proposed sampling and generation algorithms are evaluated on real world social network graphs, for which they demonstrate to be effective.

1. Introduction

With steadily increasing size and popularity, online social networks (OSNs) such as Facebook, Twitter, and Google+ have drawn the interest of the scientific community. Analyzing the structure and properties of these networks allows research in various fields. By studying social behavior and finding patterns in the networks' structure, it is possible to acquire knowledge about requirements and parameters for future networks and applications. Additionally, OSNs have a high impact on today's users' choice and consumption of online media. Coupled with widespread availability of mobile Internet, these phenomena give rise to the scientific field of socially aware traffic management [29]. The main idea in this discipline is to utilize social information about Internet users in order to enhance existing traffic management strategies. For example, information from OSNs about users' interests allows for improved caching solutions. However, complete social networks are seldom available due to privacy restrictions and the OSN providers' reluctance to publish the core of their business data. Furthermore, running complex algorithms on social network graphs in the order of magnitude of hundreds of millions of nodes is usually infeasible due to time and resource constraints.

Graph sampling techniques address the latter issue by examining only a representative subset of a given graph. Formally, the task of deriving a node sample from a given graph $G = (V, E)$ with node set V of size n and edge set E can be defined as finding a subset of nodes $V_S \subseteq V$ whose topological information can be used in order to reliably estimate various properties of G. There are two main quality requirements for sampling strategies. First, the generated sample has to be unbiased. That is, the expected value of the sampled data and the actual value of the estimated parameter are equal. Second, the minimum amount of samples required for reliable results should be low.

Unfortunately, state-of-the-art graph sampling techniques are limited to the topological analysis of huge graphs whereas socially aware traffic management requires additional information on user attributes like interests, geographic location, and age. Retrofitting these algorithms with attribute sampling capabilities implicitly assumes the independence of attributes and topology and can provide inferior results. Therefore, this work transfers ideas from graph sampling to graphs with node attributes by proposing a joint sampling of structure and attributes, which takes possible dependencies between

* Corresponding author. Tel.: +49 931 3188475; fax: +49 931 3186632.
E-mail addresses: seufert@informatik.uni-wuerzburg.de (M. Seufert), stanislav.lange@informatik.uni-wuerzburg.de (S. Lange), tobias.hossfeld@uni-due.de (T. Hoßfeld).

[1] Now at: University of Duisburg-Essen, Chair of Modeling of Adaptive Systems, Essen, Germany.

http://dx.doi.org/10.1016/j.comcom.2015.07.023
0140-3664/© 2015 Elsevier B.V. All rights reserved.

topology and attributes into account. The resulting sampling mechanism provides unbiased and reliable estimates of joint topological and attribute based properties of social network graphs in a resource efficient fashion. As an application of this sampling approach, a graph generation method [8] is augmented to use a collected sample for generating synthetic social network graphs, which show joint structure and attribute properties similar to the original graph. In order to quantify this similarity, measures were developed, which assess similarity not only with respect to topology, but also take attributes into account.

Thus, the contribution of this work is threefold:

- Existing sampling algorithms are extended to node attributes
- A novel sampling algorithm is proposed which allows for joint capturing of structure and attribute characteristics
- A graph generation method is presented that reproduces topology and attribute related properties of the original graph based on sampling

Therefore, this work is structured as follows. Section 2 covers relevant related work on attribute sampling and graph generation, and describes the used social network data sets. Section 3 introduces the sampling mechanism and presents results. Topological graph similarity measures are extended to additionally assess attribute related similarity in Section 4. A method to generate synthetic social network graphs from a node sample with attributes is proposed in Section 5. The performance of the algorithm is evaluated for different social network graphs and attributes. The results are discussed and an outlook on future work is given in Section 6.

2. Related work

2.1. Graph sampling

Numerous approaches have emerged since graph sampling became a relevant scientific topic. We revisit state-of-the-art graph sampling algorithms that are classified into three categories, namely, Uniform Node Sampling (UNI), Breadth First Search (BFS), and random walks (RW). Though primarily focused on sampling topological graph properties, these algorithms provide a solid foundation for the design of novel sampling algorithms. Due to extensive research and several performance benchmarks [17–19], they have proven properties and behavior and are also well-established in practice.

In the context of UNI, a given amount of n_S nodes is drawn at random from the original graph's set of nodes V. While this procedure guarantees an unbiased sample, it is not practical in most situations due to several possible restrictions. These include sparse ID spaces where multiple queries may be required in order to obtain a single sample, or even completely unknown ID spaces where no information about the domain of user IDs is available. Thus, UNI is considered as reference for the theoretical best case. Although BFS and related methods like depth first search [17] and snowball sampling [12] were used for various sampling tasks in the past [24,25], current research suggests avoiding these methods due to a bias towards nodes with high degree [17,18]. Additionally, this bias is graph specific and no mechanism for correcting this bias has been developed yet. Recent graph sampling mechanisms rely on random walks [11,20,27], a family of algorithms that require only the basic operation of querying a node for its set of neighbors. With the possibility to exactly quantify the node degree bias encountered in random walks, techniques for correcting this bias have been developed and allow collecting samples that are unbiased with respect to topology. Most commonly used representatives include the Metropolis Hastings Random Walk (MHRW) and the Re-Weighted Random Walk (RWRW). Based on the Metropolis Hastings algorithm [23], MHRW is a rejection sampling technique that corrects the bias on the fly. Its applications range from the analysis of P2P networks [26,31] to that of directed [34] and undirected [7] OSNs. In contrast, RWRW first performs a biased RW and then applies the Hansen–Hurwitz estimator [10] to the degree distribution observed in the sample. By dampening the occurrence probabilities of high degree nodes, this yields an unbiased distribution. The main advantage of RWRW over MHRW is that RWRW avoids spending a large portion of its sampling budget on rejections. In the case of MHRW, around 55% of iterations are rejections [7]. Therefore, on average, MHRW's resulting set of sampled nodes not only consists of fewer unique nodes, but also stems from a shorter walk. A generalization of the RWRW algorithm is presented in Section 3 and is the basis of the developed sampling algorithm. It allows for estimation of the two-dimensional distribution of node degrees and attribute values. While the literature also offers techniques for sampling from dynamic, time dependent graph streams [1], our proposed method works with static graphs. The main reason for this behavior is that typical methods involving dynamic graphs require operations like drawing a graph's edges uniformly at random which is not possible in real world OSN graphs. Furthermore, algorithms for estimating a graph's size have been proposed [13]. However, we assume that the graph's size is part of the input.

2.2. Graph generation

Modeling real networks is an important branch of science with many applications, including the analysis of biological and social systems. When a model is able to use a real graph to consistently generate synthetic graphs that capture the majority of the original graph's properties, its resulting graphs can be used as input for algorithm benchmarks and simulations, or as a means of anonymizing crawled data before publication. This section outlines three state-of-the-art models that can be used to generate synthetic graphs given either the full input graph or even just a sample of its node set.

Exponential Random Graph Models (ERGMs) [16,30,35] constitute a family of statistical models whose goal is to reveal dependencies in the process of edge creation in networks. This is achieved by quantifying the importance of various graph statistics that summarize structural patterns in the graph. ERGMs are often used for characterizing social networks [4,6,28]. Additionally, these models allow generating synthetic graphs once model parameters have been estimated. However, ERGMs require complete information on the input graph and current implementations' time complexity prohibits the analysis of graphs whose size exceeds a few thousand nodes [36], thus excluding most real world OSNs.

In [14], the Multiplicative Attribute Graph Model (MAG), a generative model for graphs with categorical node attributes, is proposed. The basic idea is that the probability of two nodes being involved in an edge depends on the nodes' pairwise combinations of attribute values. By quantifying the probability of edge formation for each possible combination of attribute values, various relationships like homophily, heterophily, or the tendency to seek connections to a specific attribute value can be expressed. Given the original graph, model parameters representing the aforementioned probabilities can be estimated. Based on these parameters, the model is capable of generating synthetic graphs with realistic topological and attribute related properties [15]. Unfortunately, MAGs also require the full input graph G for parameter estimation. As in the case of ERGMs, this is the factor that makes the approach unsuitable for this work's goals of analyzing real world OSNs and generating similar networks based on samples.

The authors of [8] present a method for generating a topology similar to the one of a real world graph without requiring full knowledge of that graph. The input consists of a node sample collected during a random walk on the graph of interest. Main aspects to reproduce are the joint degree distribution (JDD), basically an edge count for each type of degree–degree combination, as well as the average clustering coefficient per node degree. For this purpose, values of the aforementioned measures are derived from the collected sample and are forwarded to an algorithm that generates graphs whose measures approximate the targeted values.

The notion of 2.5K-Graphs stems from the work on dK-series [22] that describes graph models that contain increasing degrees of information on the graphs' structure and have increasing complexity:

0K captures the average node degree.
1K captures the node degree distribution.
2K captures the joint degree distribution.
3K captures two distributions: counts of wedges (node chains of length three) and counts of triangles among triples of node degrees k_1, k_2, k_3.

For a graph $G = (V, E)$ with node set V and edge set E, the 2K model specifies the JDD as defined in Eq. (1). For each pair of node degrees k and l, it returns the number of edges between nodes of those degrees. The sets V_k and V_l denote the subsets of V that consist exclusively of nodes of degree k and l, respectively.

$$\text{JDD}\,(k, l) = \sum_{a \in V_k} \sum_{b \in V_l} 1_{\{\{a,b\} \in E\}} \tag{1}$$

Unfortunately, the JDD does not contain information on any sort of clustering or centrality, thus making the 2K not expressive enough to model real networks. 3K, on the other hand, does contain such information, but there is currently no efficient algorithm for the 3K model. Therefore, 3K is not suitable for the analysis of real networks either. The 2.5K model tries to bridge this gap by adding the average clustering coefficient per node degree $\bar{c}(k)$ to the 2K approach, thus maintaining the efficiency of 2K but also providing a centrality measure in order to achieve more realistic results.

The first step in the 2.5K framework consists of performing a random walk on the graph to replicate. The random walk does not only record each visited node's ID and degree but also its adjacency list. Later, it is possible to induce edges in the traversed graph by checking different nodes' sets of neighbors for intersections. By applying the Hansen–Hurwitz estimator to the sampled set of nodes, the node degree bias of the random walk is corrected and estimates for the JDD and $\bar{c}(k)$ are derived. After a postprocessing step that includes smoothing of the JDD and ensures that the resulting JDD is actually realizable by a real graph, the generation phase of the algorithm is initiated.

In this phase, the algorithm first creates a set of nodes that follows the degree distribution encoded in the JDD. Each of these nodes has a target degree and is therefore considered to have "stubs" that can be connected to edges. In the second step, these stubs are connected. Two important constraints are enforced during this procedure. First, an edge may only be added if the resulting graph does not exceed the edge count defined in the JDD. Second, edges are added in a greedy fashion that strives for a high clustering coefficient. The latter part is important for runtime purposes. In the final steps, double edge swaps are performed in a Markov Chain Monte Carlo (MCMC) fashion. These swaps guarantee that the JDD is preserved while $\bar{c}(k)$ approaches the targeted distribution.

The results presented in the paper compare the 2.5K graphs with real and generated graphs from the 2K model and show far better performance while being fast enough for practical use on large real world graphs. Graph statistics in this comparison include the distributions of node degree and average neighbor degree per node degree, the average clustering coefficient per node degree, and the joint degree distribution.

Considering the goals of this work, the main issue with the 2.5K approach is that its only interest lies in reproducing the input graph's topology while ignoring node attributes. Another concern may be that the output graphs need to have roughly the same size as the input graph as the JDD matrix contains absolute integer values that lead to varying edge densities depending on the size of the output graph. The approach presented in Section 3.2 tries to generalize the JDD estimation in order to support node attributes.

In [9], two graph generation algorithms capable of reproducing different characteristics of a given input graph are presented. Both algorithms guarantee achieving the same joint degree distribution as the input graph. Additionally, the first algorithm aims at achieving an average clustering coefficient that is close to that of the input graph while the second is directed towards the joint distribution of node degrees and attribute values. Similarly to the approach presented in this work, the algorithm uses an extension of the JDD that captures attribute related properties, referred to as JDAM (joint degree and occurrence of attributes matrix), which is a concept already used in earlier work [5,21]. In contrast to [9], our proposed graph generation mechanism strives towards both goals simultaneously, i.e., joint degree and attribute distribution as well as average clustering coefficient. Furthermore, our algorithm deals with the challenge of having an incomplete view of the input graph as its input consists of a random walk node sample rather than the entire graph.

2.3. Data set description

In order to conduct performance evaluation of various graph algorithms presented in this work, realistic input data are required. As the envisaged application of the algorithms is in the field of social networks and socially aware traffic management, we focus on publicly available graphs of OSNs. The use of real world input graphs ensures representative results. For the experiments performed in this work, two data sets containing topological and attribute related information were used. The Pokec data set[2] published in [32] contains the whole network of Pokec[3], a popular Slovakian OSN with over 1.6 million users. User profiles feature information on age, interests, gender, and several other attributes. With over 1.6 million nodes, it is the biggest network studied in this work. Additionally, a collection of 100 Facebook subgraphs[4] published in [33] is analyzed. These subgraphs cover different American colleges and universities whose sizes range from 760 to 41,000 nodes. They include information on students' gender, class year, major, high school, and dormitory. Details regarding the graphs' topological and attribute related properties can be found in the corresponding publications. Various characteristics of the subset of graphs that is used for evaluating the proposed sampling and graph generation algorithms are listed in Table 3.

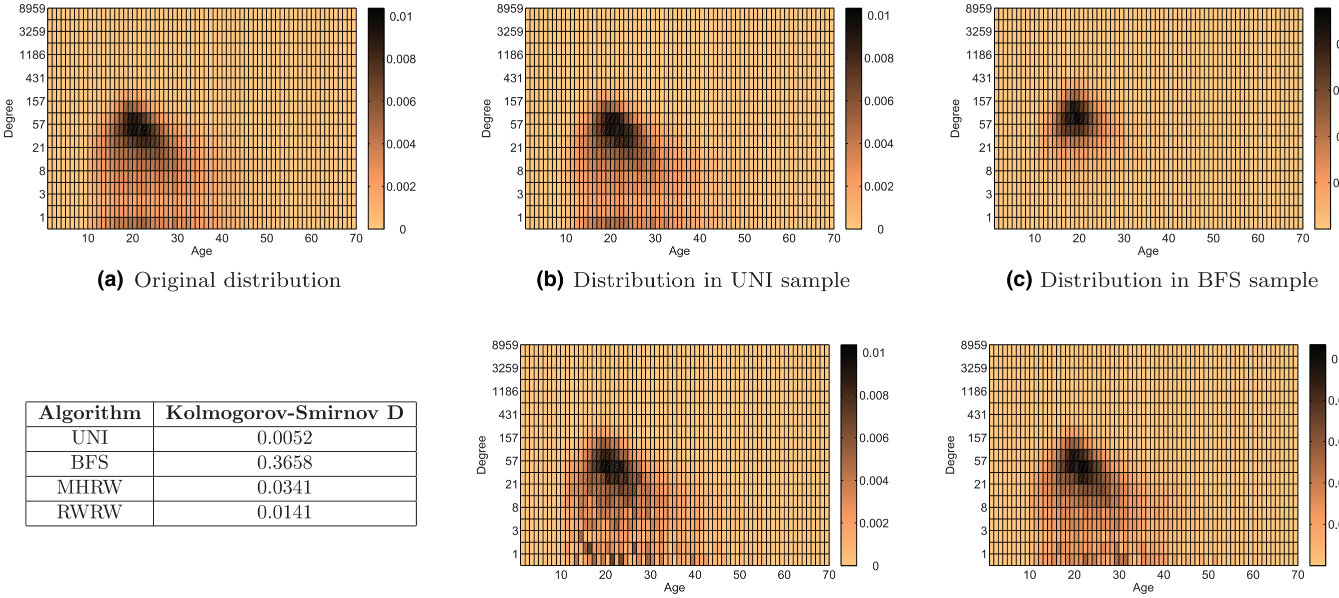
3. Proposed sampling approach

3.1. Extending existing sampling algorithms to attributes

When designing sampling algorithms that take into account both the graph's topology and its attribute values, a first approach could be extending existing graph sampling algorithms to collect attribute values when visiting a node. While UNI, BFS, and MHRW can be intuitively extended to estimate the two-dimensional degree attribute distributions, the re-weighting process of RWRW needs some modification. The resulting estimator for the two-dimensional attribute-degree distribution is presented in Eq. (2), which indicates the corrected probability for a node v with degree $\text{deg}_v = d$ and attribute value $\text{att}_v = a$. As the random walk bias does not depend on the node attribute, the Hansen–Hurwitz estimator of the one-dimensional RWRW can be adopted.

$$P(\text{deg}_v = d, \text{att}_v = a) = \frac{\sum_{w \in V_S} \frac{1_{\{\text{deg}_w = d, \text{att}_w = a\}}}{d}}{\sum_{w \in V_S} \frac{1}{\text{deg}_w}} \tag{2}$$

[2] https://snap.stanford.edu/data/soc-pokec.html
[3] http://pokec.azet.sk/
[4] https://archive.org/details/oxford-2005-facebook-matrix



(a) Original distribution

(b) Distribution in UNI sample

(c) Distribution in BFS sample

Algorithm	Kolmogorov-Smirnov D
UNI	0.0052
BFS	0.3658
MHRW	0.0341
RWRW	0.0141

(d) Values of the two-dimensional Kolmogorov-Smirnov D statistic

(e) Distribution in MHRW sample

(f) Distribution in RWRW sample

Fig. 1. Comparison between the original two-dimensional degree age distribution in the Pokec graph and those observed by the extended sampling algorithms.

Fig. 1a presents the two-dimensional joint distribution of age and node degree for the Pokec graph. Nodes' age and node degree value combinations are aggregated in bins in order to achieve a smooth plot. Each bin represents the combination of an individual age value with a range of degree values. In the figure, each rectangular cell corresponds to a bin with age values on the x-axis and degree margins on the y-axis. Analogous to logarithmically scaling the y-axis, the bin width with respect to node degree increases logarithmically in order to fit the whole range of possible degree values. The color of each cell indicates the probability of occurrence for the respective range of combinations. Fig. 1b, c, e, and f illustrate results of UNI, BFS, MHRW, and RWRW for a sample size of 100,000 on the Pokec graph. Additionally, the performance of these algorithms with respect to estimating the input graph's two dimensional degree attribute distribution is evaluated by calculating the Kolmogorov–Smirnov D statistic between the two dimensional degree attribute distribution observed in the input graph and the distributions estimated by the sampling algorithms. This statistic indicates the supremum of the distance between the respective empirical cumulative distribution functions, which is sensitive to both their locations and shapes and is thus an appropriate metric for the similarity of the distributions. While UNI yields a distribution that is barely distinguishable from that of the original graph ($D = 0.0052$), it does not come as a surprise that BFS does not perform well in the joint sampling scenario ($D = 0.3658$) as it has already been shown to be no viable approach even in the context of topological graph sampling. Also MHRW ($D = 0.0341$) clearly struggles with accuracy, especially in the lower half of the figure where probabilities of combinations involving low degree values are plotted. This phenomenon can be explained with the discrepancy introduced by the rejection procedure of MHRW. RWRW ($D = 0.0141$), on the other hand, shows the best performance of the three sampling algorithms that are feasible in practice.

Although the two-dimensional distribution can be reproduced quite accurately by RWRW, no structural dependencies can be captured by the existing sampling methods. Therefore, we propose a novel sampling approach based on RWRW, which adds an estimate for the joint two-dimensional degree attribute distribution. This estimate lays the foundation for a new graph generation algorithm, which allows for the reproduction of topological and attribute related properties of the original graph.

3.2. Sampling method

The sampling algorithm developed during this work is based on the 2.5K approach of Gjoka et al. [8] that is outlined in Section 2 and works as follows. First, a random walk is performed on the original, unknown graph. This random walk not only collects node degrees and associated attribute values, but also saves each visited node's ID and a list of the IDs of its neighbors. Due to the multitude of possible attribute-degree combinations and therefore often very low number of node occurrences per type, binning is applied with respect to the node degree. Afterwards, the random walk's node degree bias is compensated by applying the Hansen–Hurwitz estimator to the created bins. These re-weighted bins can be used to estimate the probability distribution of attribute-degree combinations in the original graph. Additionally, the adjacency lists of sampled nodes can be used to induce edges beyond those traversed by the random walk. Therefore, an estimate of the joint degree attribute distribution (JDAD), $\widehat{JDAD}$, can be derived. Intuitively, the JDAD can be thought of as a distribution of edges in the graph. For each edge type defined by the degrees and attribute values of involved nodes, the JDAD returns the number of edges of this type in the graph. A formal definition of the JDAD is provided in Eq. (3), where V_{ij} denotes the subset of nodes that have node degree i and attribute value j. The JDAD is conceptually similar to the JDAM proposed in [9], which provides the number of edges for a tuple of degree and attribute pairs. In contrast, the JDAD maps a set of pairs, which is more appropriate in the context of undirected graphs.

$$JDAD(\{(i,j),(k,l)\}) = \sum_{v\in V_{ij}} \sum_{w\in V_{kl}} 1_{\{\{v,w\}\in E\}} \tag{3}$$

Analysis of $\widehat{JDAD}$ allows estimating the assortativity of attributes of interest as well as edge densities for observed edge types. Details of the binning, re-weighting and edge induction steps are presented in the following.

The binning mechanism works as follows. After setting a constant bin width w_b, the observed degree range of each attribute is divided into bins with width w_b. Each node v is assigned to a bin $bin(v)$ depending on its node degree and attribute value. Recommended values for the bin width are 4–6. This results in sufficiently filled bins while retaining enough accuracy with respect to observed node degrees. After binning, an aggregated JDAD, $JDAD_{bin}$, can be defined according to Eq. (4). Given two bin identifiers i and j, $JDAD_{bin}(\{i, j\})$ denotes the number of edges in G that are present between nodes inside these bins.

$$JDAD_{bin}(\{i,j\}) = \sum_{\substack{v \in V \\ bin(v)=i}} \sum_{\substack{w \in V \\ bin(w)=j}} 1_{\{\{v,w\} \in E\}} \tag{4}$$

In OSNs, not all users make all their information publicly available or may choose to provide incomplete profile data. Thus, in practice, sampling algorithms come across users with unknown attribute values. In order to cope with such cases, the proposed sampling algorithm maps unset attribute values to a unique reserved attribute value (e.g., −1, N/A). On the one hand, this allows statements about the percentage of users with unavailable attribute information. On the other hand, the sampling budget spent on fetching the user's data is not wasted as topological properties still contribute to the overall estimates, regardless of attribute values.

While the node sequence $V_S = (s_1, \ldots, s_n)$ returned by a random walk of length n yields at most $n-1$ edges, namely those traversed by the random walk, a multitude of edges can be induced when adjacency lists of visited nodes are also taken into account. Edge induction exploits the fact that the original graph G contains an edge $\{u, v\}$ iff u is part of v's set of neighbors $\mathcal{N}(v)$. However, if v is sampled, not all elements of $\mathcal{N}(v)$ are necessarily visited by the random walk. Thus, degree and attribute information is only available for pairs $\{u, v\}$ where both u and v are in the sample and only edges between such nodes can be added to the JDAD. Because of the random walk's bias towards high degree nodes, the distribution of induced edges also has an inherent bias. In order to reduce effects caused by this bias, the idea of using a safety margin M as proposed in [8] is employed. Ignoring induced edges that result from checking the adjacency lists of nodes that are closer than M positions in the node sequence returned by the random walk decreases introduced bias. It is recommended to use values in the range $10 \leq M \leq 100$ [8].

As in the case of the two-dimensional degree attribute distribution, the number of possible edge types defined by the degree and attribute values of involved nodes is very large while individual numbers of occurrence are low. For this reason, the same binning procedure is applied to the list of induced edges, resulting in edge types being defined by pairs of bin IDs instead of pairs of degree-attribute combinations. After this conversion, the lists of traversed and induced edges are aggregated into an estimate of a joint bin distribution $\widehat{JDAD}_{bin}$. For each observed edge type $\{i, j\}$, the value $\widehat{JDAD}_{bin}(\{i, j\})$ represents the ratio between the number of edges of this type and the maximum possible number of such edges, given the sequence of sampled nodes V_S. Eq. (5) defines this value formally.

$$\widehat{JDAD}_{bin}(\{i,j\}) = \frac{\displaystyle\sum_{\substack{s_k, s_l \in V_S, \\ \text{s.t. } bin(s_k)=i, bin(s_l)=j, \\ |k-l|=1 \vee |k-l|>M}} 1_{\{\{s_k, s_l\} \in E\}}}{\displaystyle\sum_{\substack{s_k, s_l \in V_S, \\ \text{s.t. } bin(s_k)=i, bin(s_l)=j, \\ |k-l|=1 \vee |k-l|>M}} 1} \tag{5}$$

As entries in $\widehat{JDAD}_{bin}$ represent density measures for each edge type, they can be used to calculate an estimate of the original graph's assortativity. Additionally, scaling of this distribution with respect to the output graph size is used in the context of graph generation in order to determine goal values for the number of edges per type in the output graph.

3.3. Results

As explained in Section 3.1, existing graph sampling algorithms can easily be extended with attribute sampling capabilities. Fig. 2 quantifies the performance of these algorithms with respect to estimating two dimensional degree attribute distribution of the Bucknell University Facebook subgraph in terms of the Kolmogorov–Smirnov D statistic. Additionally, the performance of UNI and the proposed sampling algorithm is presented. The x-axis indicates the sampling budget in terms of the original graphs' sizes, while the calculated Kolmogorov–Smirnov D statistic is provided on the y-axis. In addition to the mean D value across 50 repetitions indicated by the bars' heights, whiskers denote 90% confidence intervals. For various values of the bin width parameter, the proposed graph sampling algorithm was applied to different social network graphs. The results indicate that minimum D values are achieved with bin widths of 4 to 6. As discussed in Section 2, BFS and RW suffer from a bias towards high degree nodes which results in a skewed degree distribution among the collected nodes. Thus, it is not surprising that these algorithms perform significantly worse than the presented alternatives. With increasing sampling budget, BFS's performance improves slightly. This can be explained by the fact that the BFS algorithm visits each node at most once and thus is guaranteed to visit lower degree nodes when a higher sampling budget is available. UNI is the theoretical best case as it draws random sample pairs from the original distribution. This is reflected by UNI having the lowest D score among all sampling strategies. For every sampling budget, the proposed algorithm outperforms MHRW, which is statistically shown by a two-sample t-test with p-values of 0.0028, 0.0002, 0.0038, 0.0026 for the different fraction values 0.1, 0.2, 0.3, 0.5, respectively.

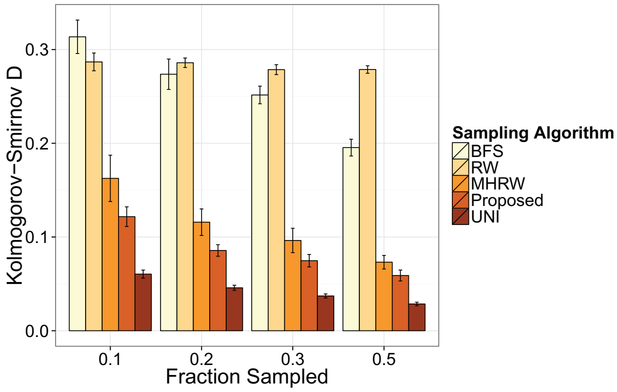


Fig. 2. Performance comparison of different sampling strategies with respect to the two dimensional degree attribute distribution. Algorithm parameters: attribute *dormitory* and Bucknell University Facebook subgraph as input graph.

The above presented results for a rather small social network graph with only 3824 nodes showed that the algorithms require sample sizes beyond 20% of the original graph's number of nodes in order to achieve a good performance with respect to the Kolmogorov–Smirnov distance. However, results obtained from sampling huge graphs like the Pokec graph show that a sampling budget in the order of magnitude of 2% to 5% of the original graph's size is sufficient in order to produce a reliable estimate. This behavior is presented in Fig. 3 where the Kolmogorov–Smirnov distance between the original $JDAD_{bin}$ and its sample based estimate, which is the foundation for the proposed graph generation mechanism, is displayed for various graphs and sample sizes. While the x-axis shows the sampling budget, the y-axis indicates the aforementioned Kolmogorov–Smirnov distance with respect to the $JDAD_{bin}$. For the four Facebook subgraphs Haverford, Rice, Stanford, and Rutgers, the *dormitory* attribute was sampled. In the case of Pokec, *age* was chosen. Three levels of performance corresponding to the graphs' size can be identified in the plot. First, the small graphs Haverford and Rice with 1446 and 4083 nodes which result in the highest Kolmogorov–Smirnov distances. Second, medium sized graphs Stanford and Rutgers containing 11586 and 24568 nodes have significantly better performance values. Finally, for

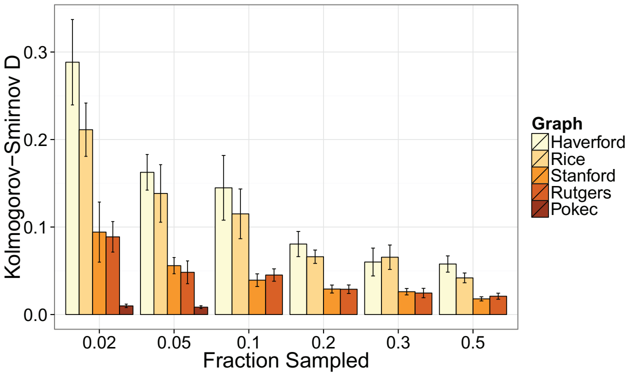


Fig. 3. Influence of the sampling budget on the Kolmogorov–Smirnov distance between $JDAD_{bin}$ and $\widehat{JDAD}_{bin}$.

the Pokec graph consisting of more than 1.5 million nodes, a sampling budget of 2% is sufficient for achieving D values below 0.01. These observations suggest that in the context of huge real world graphs, the resource efficiency achieved via sampling increases significantly.

Additional investigations show a graph and attribute independent relationship between the absolute sampling budget and the resulting Kolmogorov–Smirnov distance. Fig. 4 presents an aggregated view on samples of *dormitory*, *major*, and *year* attributes from the four Facebook subgraphs Haverford, Rice, Stanford, and Rutgers, and the *age* attributes from the Pokec graph. It displays a scatter plot of absolute sampling budgets on the x-axis alongside the corresponding D values on the y-axis. Furthermore, locally weighted polynomial regression yields the smoothed curve with a shaded area indicating the 95% confidence intervals. When both axes are logarithmically scaled, the fitted curve resembles a straight line which in turn corresponds to a power law distribution. Thus, for the investigated social network graphs, rather than relative sampling budget, the absolute sampling budget seems to be a key performance indicator of the proposed algorithm. Hence, following figures present absolute values.

Depending on the intended use case, different user attributes may be of interest when performing attribute sampling and later graph generation tasks. Fig. 4 already showed that a large sampling budget provides accurate samples almost independent of graph size and attribute type. However, for small OSN graphs with low numbers of sampled nodes some challenges may arise. Fig. 5 provides an overview of the attribute sampling algorithm's performance when applied to the same topology but different attributes. In 20 experiment repetitions per configuration, the graph sampling algorithm was applied to the small Haverford graph (1446 nodes) using the attributes *dormitory*, *major*, and *school year*. While *dormitory* and *major* are categorical attributes, *school year* is a numerical attribute with integer values. With the sampling budget on the x-axis and the Kolmogorov–Smirnov D statistic on the y-axis, the plot contains results for all three attributes. Each bar color represents one attribute. The mean D score of all repetitions is indicated by the bars' height, and whiskers depict the 90% confidence intervals. Similar to previous observations, similarity correlates with the sampling budget. However, the *school year* attribute requires a significantly higher sampling budget in order to reach reasonable distance values. An explanation for this behavior is that the characteristics of this attribute distribution are different from the two categorical attribute distributions. For example, the *school year* attribute has a high fraction of values which only occur very rarely. Therefore, the sampling algorithm needs to visit a larger amount of nodes in order to observe enough representatives and provide reliable estimates. With a sampling budget of 289 (20%), the mean D value drops below 0.1 for all attributes.

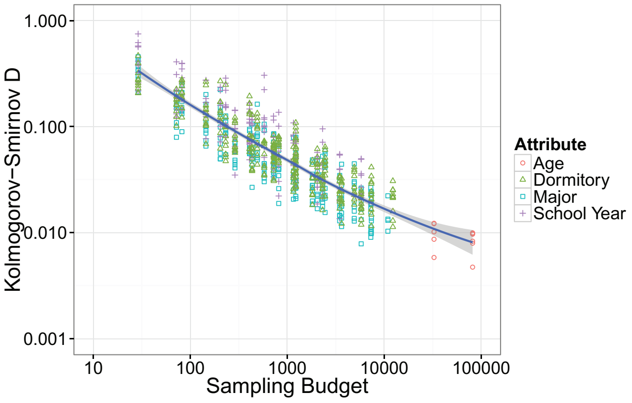


Fig. 4. Graph and attribute independent relationship between sampling budget and the Kolmogorov–Smirnov distance between $JDAD_{bin}$ and $\widehat{JDAD}_{bin}$.

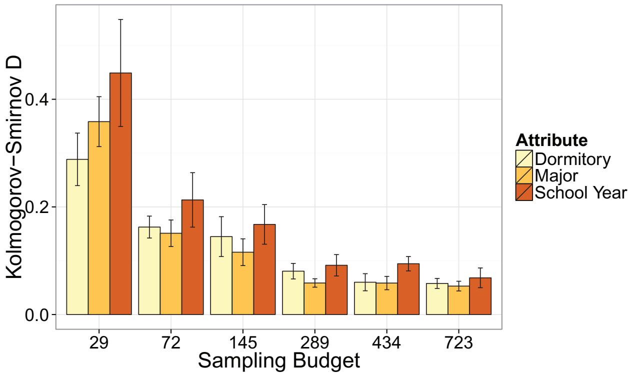


Fig. 5. Influence of the sampling budget on the Kolmogorov–Smirnov D values between $JDAD_{bin}$ and $\widehat{JDAD}_{bin}$ for different node attributes. Algorithm parameters: Haverford College Facebook subgraph and bin width 4.

On a machine equipped with an Intel Core i7 4770 CPU at 3.40 GHz and 16 GB of RAM, the sampling and $JDAD_{bin}$ estimation procedure for the Facebook subgraphs is completed within few seconds. In the case of the large Pokec graph with more than 1.6 million nodes, runtimes in the order of magnitude of 1 h are observed.

To sum up, using the proposed algorithm allows the same level of accuracy as MHRW while investing a lower sampling budget, or a higher level of accuracy than MHRW while investing the same sampling budget. Additionally, the results confirm the intuition that increasing the sampling budget results in better performance. As this effect is mainly depending on the absolute number of sampled nodes, only for small social network graphs a larger sampled fraction is needed. Additionally the properties of the sampled attributes have to be taken into account in this case. For huge graphs, on the other hand, small sampling fractions are sufficient to obtain accurate samples almost independent of graph and attribute properties.

4. Graph similarity

In order to evaluate and compare the performance of algorithms that generate synthetic graphs based on a given input graph, it is necessary to be able to quantify the output graph's similarity to the input graph. This section presents approaches for assessing topological and attribute related similarity between graphs.

4.1. Attribute based NetSimile

The comparison of multidimensional graph property distributions (e.g., by the Kolmogorov–Smirnov test) quickly becomes computationally expensive and does not provide insights into complex graph characteristics. Therefore, dedicated graph similarity measures like NetSimile [2] were designed specifically to assess graph similarity based on a variety of topological characteristics. The key idea of the algorithm is to extract node level features from each graph, aggregate these features in five ways (namely, mean, median, standard deviation, skewness, and kurtosis), and return a size independent, graph specific signature vector for each graph. By applying a distance measure to the resulting vectors, the graphs' similarity can be quantified. Unfortunately, the basic version of NetSimile is restricted to topological comparisons. Thus, additional attribute based features are introduced in order to cope with graphs that have node attributes. Following the idea of the established NetSimile measure, the developed extension, NetSimile$_{\mathrm{Att}}$, does not rely on a single attribute related property. Instead, differently aggregated statistics which represent diverse characteristics are taken into account simultaneously.

The developed NetSimile$_{\mathrm{Att}}$ measure contains four node centric attribute based properties. Before this measure can be applied, the graphs to be compared need to undergo a preprocessing step that converts them to graphs with edge weights. Based on the attribute values att_i and att_j of nodes involved in an edge $\{i, j\}$ and whether the attribute in question is categorical, continuous, or discrete, the edge's weight is determined. If the attribute is categorical, the edge weight $w_{\{i,j\}}$ is defined as $w_{\{i,j\}} = 1_{\{\mathrm{att}_i=\mathrm{att}_j\}}$. On the other hand, if the attribute is continuous or discrete, the edge weight is defined as $w_{\{i,j\}} = e^{-|\mathrm{att}_i-\mathrm{att}_j|}$. This exponentiation ensures that in either case, edge weights are normalized in the range [0; 1] where a value close to 1 indicates a homophilous relationship between the edge's nodes and a value close to 0 indicates a heterophilous relationship between them, respectively.

The following four node centric properties are analyzed by NetSimile$_{\mathrm{Att}}$:

Mean node weight For each node v, the mean weight of its edges is calculated. This statistic is defined as $\bar{w}_v = \frac{1}{\deg_v}\sum_{u\in\mathcal{N}(v)} w_{\{u,v\}}$ and quantifies the similarity between a node and its neighbors.

Mean neighbor weight For each node v, this property captures the mean value of $\bar{w}_u$ among all of v's neighbors u. Doing so extends $\bar{w}_v$ by an additional hop and thus analyzes the degree of homophily found in the two hop neighborhood of v.

Egonet edge homophily The sum of edge weights in a node's egonet (i.e., the subgraph induced by $v \cup \mathcal{N}(v)$) is calculated. Summation allows making a statement about both the size of anode's egonet and the level of similarity between included nodes.

Egonet neighbor homophily Similar to NetSimile's count of egonet neighbors, the sum of potential edge weights between node v and neighbors of its egonet u is calculated. Again, this extends the previous property by a hop and allows for a deeper insight into a node's surroundings.

Like in the case of plain NetSimile, each of these statistics is computed for every node in the compared graphs. Then, the five aggregation functions are applied to the resulting vectors which finally yields the graphs' signature vectors. The NetSimile$_{\mathrm{Att}}$ value between the input graphs is defined as the Canberra distance between these signature vectors. Low values indicate a high level of similarity while high values denote dissimilarity between graphs. In particular, the NetSimile and NetSimile$_{\mathrm{Att}}$ measures are zero when the input consists of two identical graphs. Section 4.3 shows examples of typical values for similar and dissimilar pairs of graphs and experimentally demonstrates the measure's suitability for similarity assessment.

4.2. Attribute based eigenvalue extraction

Another measure for graph similarity is briefly introduced in [2]. The Eigenvalue Extraction (EIG) algorithm stems from the area of spectral graph analysis and is based on the idea that eigenvalues can be used as an index of centrality in network structures [3]. First, given graphs are mapped to signature vectors. These vectors consist of the k largest eigenvalues of the graphs' adjacency matrices, where k is an algorithm parameter. Usually, a value of $k = 10$ is sufficient. After that the similarity measure is defined as the Canberra distance between the resulting vectors.

Unlike NetSimile's local feature extraction from each individual node, this approach extracts global features by considering the whole adjacency matrix at once. As in the case of NetSimile, EIG cannot quantify the similarity of graphs with node attributes. Therefore, using the preprocessing approach developed for NetSimile$_{\mathrm{Att}}$, EIG can also be extended to support graphs with node attributes. First, the graphs' edges $\{i, j\}$ are assigned weights $w_{\{i,j\}}$ as defined in the previous section. Then, the same procedure as in EIG is applied: the k largest eigenvalues of the graphs' weighted adjacency matrices are computed. These eigenvalues compose the graphs' signature vectors that are finally compared via the Canberra distance. This whole method of comparing graphs with node attributes is referred to as EIG$_{\mathrm{Att}}$.

4.3. Results

In this section, various methods for assessing graph similarity were introduced. These include NetSimile and EIG as well as their attribute based counterparts NetSimile$_{\mathrm{Att}}$ and EIG$_{\mathrm{Att}}$, respectively. Before the next section discusses the performance of graph generation algorithms with respect to these similarity measures, a brief overview of the domains of these measures is provided. For this purpose, the measures' behavior for pairs of social network graphs that are intuitively similar or dissimilar based on criteria like size, edge count, and attribute based assortativity is analyzed. Results from these comparisons provide a reference for assessing the performance of the graph generation algorithm.

In addition to the results presented in this section, Fig. 9 in the next section illustrates a positive correlation between the proposed NetSimile$_{\mathrm{Att}}$ measure and Kolmogorov–Smirnov distances regarding the node degree distribution, attribute value distribution, and $JDAD_{bin}$ distribution between input graphs and graphs produced by our generation algorithm. This relationship indicates that NetSimile$_{\mathrm{Att}}$ is indeed capable of capturing topological as well as attribute related graph properties in a manner which is consistent with established alternatives.

Table 1 provides values for the topological measures NetSimile and EIG for different pairs of graphs from the Facebook dataset. In the first two examples, intuitively similar graphs are compared which all have pairwise similar sizes with respect to both, node and edge count. For these, NetSimile values are below 0.16 and EIG does not exceed 0.06. The following two comparisons show dissimilar graphs. While having an almost identical number of nodes, the Simmons graph exhibits only half as many edges as the Haverford or Swarthmore graphs. As a result, NetSimile and EIG values rise beyond 0.26.

In order to grade possible values for attribute based similarity measures, Table 2 presents calculated values for NetSimile$_{\mathrm{Att}}$ and EIG$_{\mathrm{Att}}$ for different OSN graphs and attributes. Intuitively, two graphs with node attributes should be considered similar if they have a similar amount of nodes and edges as well as a similar degree of attribute

Table 1
Exemplary values of topological graph similarity measures for different Facebook subgraphs.

G_1	G_2	NetSimile	EIG
Middlebury	Vassar	0.10	0.06
Amherst	Bowdoin	0.16	0.04
Simmons	Swarthmore	0.27	0.26
Haverford	Simmons	0.32	0.26

Table 2
Exemplary values of attribute based graph similarity measures for different Facebook subgraphs.

G_1	G_2	Attribute	$\text{NetSimile}_{\text{Att}}$	EIG_{Att}
Haverford	Swarthmore	Dormitory	0.15	0.14
Amherst	Bowdoin	Dormitory	0.16	0.16
Oberlin	Wellesley	Dormitory	0.18	0.04
Amherst	Vassar	Dormitory	0.24	0.40
Amherst	Smith	Dormitory	0.31	0.44
Bowdoin	Smith	Dormitory	0.36	0.30
Haverford	$\text{BFS}^{20\%}_{Hav}$	Dormitory	0.45	0.82
Haverford	$\text{BFS}^{40\%}_{Hav}$	Dormitory	0.22	0.16
Haverford	$\text{BFS}^{80\%}_{Hav}$	Dormitory	0.08	0.10

based assortativity. The first three entries of the table provide such examples. In these cases, $\text{NetSimile}_{\text{Att}}$ does not exceed 0.18 and values of EIG_{Att} are 0.16 or less. The next three instances illustrate cases where assortativity differs by a factor of around three which directly affects the measures' values. They increase beyond 0.24 and 0.30, respectively.

The bottom part of the table presents comparisons of the Haverford graph with subgraphs of itself. $\text{BFS}^{k\%}_{Hav}$ denotes the subgraph of the Haverford graph that is based on a BFS sample of k% of its nodes. With increasing k, the subgraph approaches the original graph and thus, the similarity measures take on lower values. While $\text{NetSimile}_{\text{Att}}$ drops as low as 0.08, EIG_{Att}'s minimum is 0.10. This can be explained with the fact that EIG_{Att} has a dependency on graph size which adds to the dissimilarity already present from BFS sampling.

The examples provided in this section allow deriving thresholds for the four similarity measures which in turn help interpreting the results presented in the following section. For NetSimile, EIG, $\text{NetSimile}_{\text{Att}}$, and EIG_{Att}, these thresholds are 0.16, 0.06, 0.18, and 0.16, respectively. If the similarity value between a generated output graph and its underlying input graph falls below the corresponding threshold, the output graph is considered to be similar to the input graph.

Table 3
Comparison of avg. clustering coefficient $\bar{c}$, the assortativity a, the avg. shortest path length $\bar{l_s}$, the number of cliques $|C|$, and the avg. closeness centrality $\bar{cc}$ for original and generated graphs. Algorithm parameters: attribute *school year* and bin width 4.

Fraction	$\bar{c}$	a	$\bar{l_s}$	$\lvert C\rvert$	$\bar{cc}$
Rice ($\lvert V\rvert = 4087$, $\lvert E\rvert = 184826$)					
original	0.300	0.520	2.468	1145592	$1.00 \cdot 10^{-4}$
0.1	0.201	0.601	2.714	1048786	$4.37 \cdot 10^{-7}$
0.2	0.241	0.624	2.661	863602	$7.51 \cdot 10^{-7}$
0.3	0.258	0.575	2.636	1842283	$1.27 \cdot 10^{-6}$
0.5	0.310	0.577	2.644	2311427	$1.94 \cdot 10^{-6}$
Bucknell ($\lvert V\rvert = 3826$, $\lvert E\rvert = 158863$)					
original	0.281	0.268	2.507	522476	$1.05 \cdot 10^{-4}$
0.1	0.197	0.031	2.763	441164	$4.20 \cdot 10^{-7}$
0.2	0.283	0.085	2.756	685400	$1.40 \cdot 10^{-6}$
0.3	0.294	0.270	2.725	690691	$2.01 \cdot 10^{-6}$
0.5	0.308	0.273	2.661	1039404	$2.62 \cdot 10^{-6}$
Smith ($\lvert V\rvert = 2970$, $\lvert E\rvert = 97133$)					
original	0.289	0.157	2.498	151137	$1.37 \cdot 10^{-4}$
0.1	0.182	0.411	2.820	203916	$1.13 \cdot 10^{-6}$
0.2	0.228	0.137	2.713	282275	$2.85 \cdot 10^{-6}$
0.3	0.283	0.086	2.765	273109	$3.21 \cdot 10^{-6}$
0.5	0.333	0.197	2.780	376658	$5.35 \cdot 10^{-6}$
Haverford ($\lvert V\rvert = 1446$, $\lvert E\rvert = 59589$)					
original	0.327	0.195	2.228	475705	$3.14 \cdot 10^{-4}$
0.1	0.234	0.152	2.416	232857	$2.57 \cdot 10^{-6}$
0.2	0.290	0.158	2.303	344723	$4.23 \cdot 10^{-6}$
0.3	0.311	0.153	2.277	629996	$4.66 \cdot 10^{-6}$
0.5	0.333	0.188	2.307	637602	$5.65 \cdot 10^{-6}$

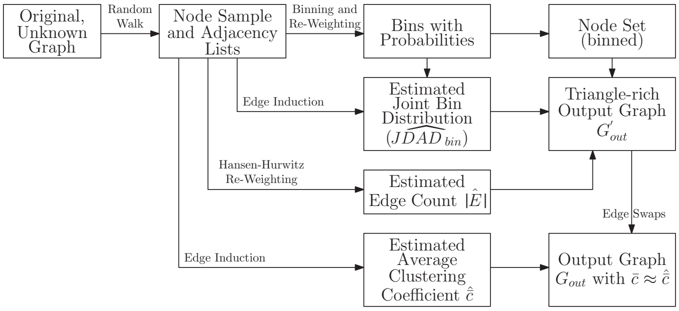

Original, Unknown Graph
Random Walk
Node Sample and Adjacency Lists
Binning and Re-Weighting
Bins with Probabilities
Node Set (binned)
Edge Induction
Estimated Joint Bin Distribution ($\widehat{JDAD}_{bin}$)
Triangle-rich Output Graph G'_{out}
Hansen-Hurwitz Re-Weighting
Estimated Edge Count $|\hat{E}|$
Edge Swaps
Edge Induction
Estimated Average Clustering Coefficient $\hat{\bar{c}}$
Output Graph G_{out} with $\bar{c} \approx \hat{\bar{c}}$

Fig. 6. Outline of the graph generation algorithm developed in this work.

5. Graph generation

5.1. Algorithm

The graph generation algorithm developed during this work is capable of creating graphs with node attributes based on a node sample collected during a random walk and the target size n of the output graph. In this work, n equals $|V|$, i.e., the size of the original graph, and is part of the input. An overview of involved mechanisms is shown in Fig. 6.

The algorithm starts off by collecting a node sample via a random walk. The sampling algorithm presented in Section 3.2 calculates occurrence probabilities for every bin $B_i = (B_{i,a}, B_{i,l}, B_{i,r})$ defined by its attribute $B_{i,\,a}$ and its degree range $[B_{i,\,l}; B_{i,\,r})$. Multiplying this bin distribution with the size of the input graph, and subsequent rounding results in node counts for every bin B_i. These counts are used to create a preliminary node set where nodes' attribute values are fixed, but whose degree range lies in the range defined by their source bin. Formally, such a preliminary node v is defined as triple $(\text{att}_v, \text{left}_v, \text{right}_v)$ consisting of its attribute value att_v and left and right degree margins left_v and right_v, respectively. Furthermore, the sampling mechanism provides $\widehat{JDAD}_{bin}$, i.e., an estimate of the joint two-dimensional degree attribute distribution, which contains the structural characteristics of the graph.

Before connecting the nodes via edges, entries of the $\widehat{JDAD}_{bin}$ need to be converted from density measures for each edge type to actual edge counts. For this, the $\widehat{JDAD}_{bin}$ is multiplied with $|\hat{E}|$, an estimate for the number of edges in the original graph. This estimate is derived by utilizing the reliable degree estimate provided by the Hansen–Hurwitz estimator and exploiting the relationship of node degree and edge count. First, the Hansen–Hurwitz estimator is applied to the node degrees observed in the random walk's node sample, which yields estimates for the occurrence probabilities p_k of each node degree k. These probabilities can be used in order to estimate the average node degree in the original graph.

The ingredients collected so far are sufficient to create an output graph whose degree-attribute distribution is similar to that of the original graph and whose edges follow the estimated $\widehat{JDAD}_{bin}$. The output graph can be constructed by iterating over all possible edges between nodes of the created node set and adding only those edges whose insertion neither violates the degree range of the involved nodes nor leads to exceeding the edge count in the $\widehat{JDAD}_{bin}$.

As explained in this section's introduction, such an algorithm ignores possible triangle structures and thus the created output graph misses key characteristics of the input graph. To address this issue, the algorithm is augmented by an estimate of the input graph's average clustering coefficient $\hat{\bar{c}}$. The estimation is based on the set of induced edges E_{ind} that is derived during the sampling process and is described in detail in the work of Gjoka et al. [8].

Now, the algorithm is extended to first greedily create triangles and thus an output graph with a high average clustering coefficient.

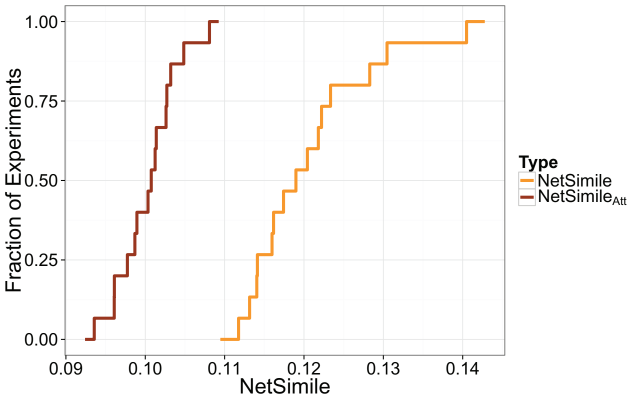


Fig. 7. Performance of the graph generation with optimal input in terms of NetSimile and NetSimile$_{Att}$ values between generated and original graph. Algorithm parameters: attribute *dormitory*, bin width 4, and Bucknell University Facebook subgraph.

This is achieved by associating every node v in the output node set with a random one-dimensional coordinate r_v, thus assigning each possible edge a distance with respect to this coordinate system and iterating through edge candidates sorted by their distance value. Following this principle, the constraints imposed by $\widehat{JDAD}_{bin}$ are still met while the resulting average clustering coefficient is increased [8].

In most cases, this construction results in an output graph with an average clustering coefficient greater than the one estimated in the previous step. Thus, in a final step, $\widehat{JDAD}_{bin}$ preserving edge swaps are performed in order to achieve an average clustering coefficient close to the estimate $\hat{\hat{c}}$. An edge swap is a rewiring procedure applied to a pair of edges $\{u, v\}$ and $\{w, x\}$ in which the edges exchange one node with each other. This results in one of two alternative pairs of edges, namely $\{u, x\}$ and $\{v, w\}$ or $\{u, w\}$ and $\{v, x\}$. A $\widehat{JDAD}_{bin}$ preserving edge swap is defined as an edge swap that does not alter entries in $\widehat{JDAD}_{bin}$. Such an edge swap can be achieved by choosing a pair of edges whose nodes' bin memberships overlap. Given $\hat{\hat{c}}$, the maximum number of iterations i_{max}, and an accuracy threshold ε, edge swaps are performed in an MCMC fashion. In each step two random edges originating from nodes with identical bin membership are chosen and the edges' destinations are replaced with each other. If the swap changes the output graph's average clustering coefficient towards its goal value, the swap is accepted. Either after the maximum number of MCMC iterations has been performed or the difference between the output graph's average clustering coefficient and $\hat{\hat{c}}$ drops below ε, the MCMC procedure halts and the output graph is returned.

The current implementation of the graph generation algorithm is limited to creating output graphs consisting of the same amount of nodes as the input graph. Scaling the output graphs to arbitrary sizes would require nontrivial changes to both, degree and edge distributions in order to achieve realistic results. This task is out of the scope of this paper and left for future work.

5.2. Results

This section investigates the influence of different parameters on the graph generation algorithm and compares its performance to that of a state-of-the-art algorithm. Fig. 7 illustrates the performance of the graph generation for both, the topology focused NetSimile measure, as well as for NetSimile$_{Att}$, which covers similarity with respect to graphs' attribute values. Both plots are based on experiments that generated synthetic social network graphs from the Bucknell University Facebook subgraph with 3824 nodes. The sampled attribute is *dormitory*, a categorical value indicating a user's residence. To avoid propagated inaccuracies from the sampling process, the synthetic graphs are generated using the original $JDAD_{bin}$. Thus, the plot presents the theoretical performance of the generation algorithm when it is running with optimal input. It shows the CDFs of NetSimile and NetSimile$_{Att}$ values after the generation of 15 synthetic graphs based on the original $JDAD_{bin}$. The NetSimile values between the original graph and the generated graphs range from 0.11 to 0.14. Additionally, NetSimile$_{Att}$ values form a steep CDF with scores always below 0.11. It follows that the generated graphs are very similar to the original graph as both similarity measures are well below the thresholds found in Section 4.3. This means, the proposed graph generation algorithm is able to accurately reproduce topological and attribute based properties based on $JDAD_{bin}$ of the input graph. Note that in order to obtain $JDAD_{bin}$, full knowledge and processing of the input graph is required, which is not feasible for larger graphs. Therefore, each graph generation is usually preceded by a sampling process in practice.

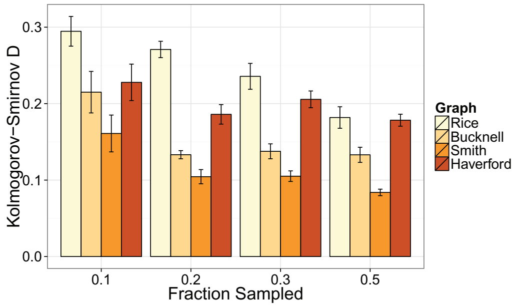


Fig. 8. Influence of the sampling budget on the Kolmogorov–Smirnov distance between original and generated $JDAD_{bin}$. Algorithm parameters: attribute *school year* and bin width 4.

The practical application performance of the graph generation algorithm is presented in Figs. 8 and 10. This means, a sampling was performed on four different graphs of the Facebook data set with four different sampling budgets and the estimates of $\widehat{JDAD}_{bin}$ were used as input to the generation algorithm. The chosen OSN graphs, namely Rice, Bucknell, Smith, and Haverford, cover a wide range of topological and attribute related characteristics. Thus, the algorithm's performance on these graphs can be used as indicator for the algorithm's overall performance. The collected attribute is *school year*. To present the results for the different graphs in a decent way, sampled fraction is used on the x-axis. However, note that the above findings on the accuracy of the proposed sampling algorithm remain valid and propagate through the generation process. This means, instead of sampled fraction, the performance of the generation algorithm also mainly depends on the absolute number of sampled nodes.

Fig. 8 shows the Kolmogorov–Smirnov distance between the original $JDAD_{bin}$ and $JDAD_{bin}$ of the generated graph. In contrast to the

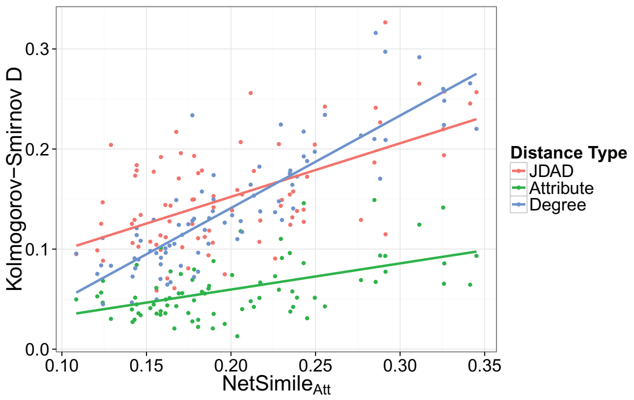
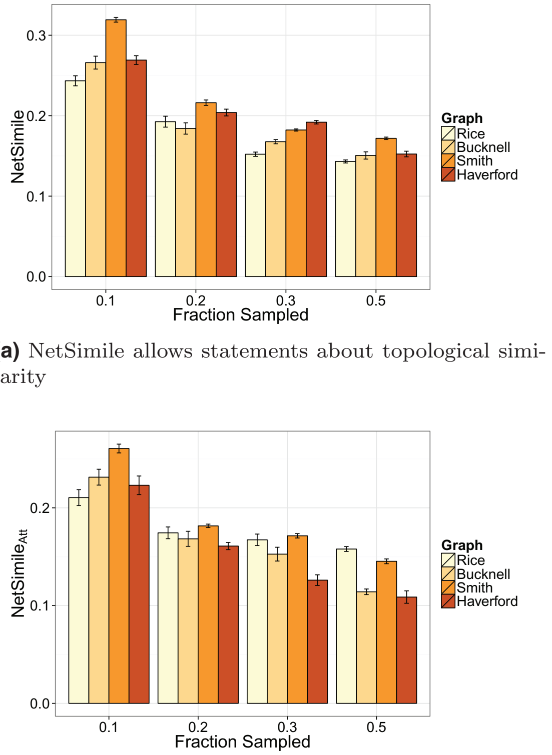


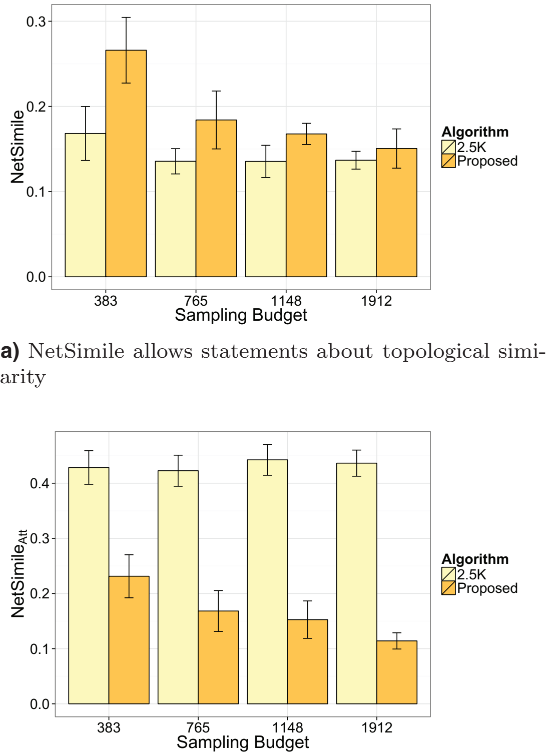
Fig. 9. Correlation of NetSimile$_{Att}$ and Kolmogorov–Smirnov distance between JDAD, attribute, and degree distributions.



(a) NetSimile allows statements about topological similarity

(b) $\text{NetSimile}_{\text{Att}}$ captures similarity with respect to the graph's attributes

Fig. 10. Influence of the sampling budget on NetSimile and $\text{NetSimile}_{\text{Att}}$ values between generated and original graphs. Algorithm parameters: attribute *school year* and bin width 4.



(a) NetSimile allows statements about topological similarity

(b) $\text{NetSimile}_{\text{Att}}$ captures similarity with respect to the graph's attributes

Fig. 11. Performance comparison between the 2.5K approach and the algorithm proposed in this work. Algorithm parameters: attribute *school year*, bin width 4, and Bucknell graph.

results reported in Fig. 5, the calculation of the distances presented in this section also takes into account the fact that the graph generation mechanism might not be able to reconstruct the estimate of $JDAD_{bin}$ exactly. Nonetheless, the value of the D statistic decreases for each graph when the available sampling budget is increased.

In order to investigate the relationship between the similarity measures proposed in Section 4 and the established Kolmogorov–Smirnov D statistic, the different distance measures between original and generated graphs are calculated and compared with each other. Fig. 9 presents the correlation between $\text{NetSimile}_{\text{Att}}$ and the Kolmogorov–Smirnov distance with respect to three different distributions. These include the single dimensional distribution of attribute values, the single dimensional distribution of node degrees, and the combined distribution represented by $JDAD_{bin}$. While the x-coordinate of each point denotes its $\text{NetSimile}_{\text{Att}}$ value, the y-coordinate represents the corresponding Kolmogorov–Smirnov D. In addition to this scatter plot, a line representing the linear best fit is added for each distribution used by the Kolmogorov–Smirnov statistic. The positive correlation between $\text{NetSimile}_{\text{Att}}$ and topological, attribute based, and joint Kolmogorov–Smirnov distances highlighted in the figure demonstrates its feasibility for the performance assessment of our graph generation mechanism. Thus, results are presented in terms of $\text{NetSimile}_{\text{Att}}$ for the remainder of this work.

In Fig. 10, the respective similarity measures NetSimile and $\text{NetSimile}_{\text{Att}}$ are shown. Again, bar height indicates the mean across 20 repetitions while whiskers represent 90% confidence intervals. Fig. 10a conveys that there is a direct relationship between an output graph's similarity to the input graph and the size of the sample the former is based on. This is not surprising as plots for the sampling algorithm revealed a similar relation. Thus, the graph generated on a more reliable estimate is also more likely to express a higher degree of similarity to the original graph. Fig. 10b summarizes the algorithm's performance with respect to its ability to replicate attribute based graph characteristics. Using the same reasoning as before, an increasing level of similarity is observed when a higher sampling budget is available. In contrast to NetSimile, however, the algorithm achieves good values of 0.18 and below for the $\text{NetSimile}_{\text{Att}}$ measure with a sampling budget starting at 30%. Additionally, it can reliably reproduce attribute characteristics of the Rice, Bucknell, Smith, and Haverford Facebook subgraphs after analyzing just 20% of their nodes.

In Table 3, other numerical graph properties of the original and generated graphs are presented. These properties include the average clustering coefficient $\bar{c}$, the assortativity a, the average shortest path length $\bar{l}_s$, the number of cliques $|C|$, and the average closeness centrality $\bar{cc}$. The values for the generated graphs represent averages based on ten generation runs. It can be seen that the clustering coefficient, which is a target metric for the generation, and the assortativity are well replicated by the generated graphs especially if the sampled fraction is high enough. However, the average shortest path lengths tend to become slightly larger and also the number of cliques and the closeness centrality are not close to the original, which shows that there is still some room for improvement of the algorithm.

While the results shown so far demonstrate performance and the influence factors of the graph generation algorithm developed during this work, Fig. 11 provides a direct comparison with a state-of-the-art algorithm. In addition to the publication introducing the mechanisms utilized in the 2.5K approach [8], a Python implementation of the graph generation algorithm is provided. Using this implementation[5], social network graphs were also generated via the 2.5K method. Fig. 11a compares the topological similarity achieved by the 2.5K approach with that achieved by the proposed mechanism. Experiments

[5] http://www.minasgjoka.com/2.5K/instructions/index.html

were conducted on the Facebook graph of the Bucknell University, which contains 3824 nodes. The x-axis presents information on the available sampling budget and the y-axis displays the mean NetSimile value calculated for the input and output graphs after 20 experiment repetitions. Whiskers attached to the bars indicate the 90% confidence intervals. Different bar colors denote the two graph generation algorithms. While the 2.5K algorithm outperforms the proposed method in terms of topological similarity in all instances, the difference becomes steadily lower until it is barely relevant beyond a sampling budget of 1148 (30%), as confidence intervals begin to overlap. The performance difference can be explained by the binning mechanism incorporated in the proposed algorithm. This mechanism adds some inaccuracy with respect to the output graph's node degree distribution in order to cope with estimation difficulties.

In its published version, the 2.5K algorithm is not designed for generating graphs with node attributes. Nonetheless, a comparison with the algorithm developed in this work can be drawn by assigning attribute values to the graphs generated by the 2.5K approach. The attributes are assigned to the output graph's nodes according to the degree specific attribute value distribution observed in the collected sample. Fig. 11b presents NetSimile$_{\text{Att}}$ values for this scenario. Due to a lack of information on the attribute distribution in edges and higher order substructures of the input graph, the 2.5K algorithm fails to produce similar output graphs. On the other hand, the algorithm developed in this work expresses a high degree of similarity as soon as a sampling budget of 1148 (30%) is available. Paired with its capability of reliably reproducing the input graph's topology, it should be favored in scenarios containing graphs with node attributes.

Using the test machine[6], the entire process of sampling, estimation, and generation takes between 20 and 40 min for the Facebook subgraphs under study. For these, the number of edges in the graph is identified as the main influence factor on runtime. In the context of large scale networks like Pokec, the time and memory requirements of the current implementation prohibit an evaluation. Future work will address these issues by investigating alternative edge insertion mechanisms in order to increase the maximum size of generated graphs.

6. Discussion

In this work, a practical methodology for efficiently estimating topological and attribute related properties of graphs with node attributes was developed. As this estimation relies on a node sample whose size is significantly smaller than that of the input graph's node set, the developed sampling algorithm can be used in order to estimate properties of huge real world graphs like online social networks. This property makes it suitable for different use cases, e.g., socially aware traffic management, where attribute related graph properties need to be computed in a fast and reliable manner. Furthermore, a mechanism for generating synthetic graphs with node attributes was designed. In contrast to previous state-of-the-art graph generation algorithms, it does not require full knowledge of the input graph in order to replicate the graph's key characteristics with respect to topology and node attributes. Thus, crawling a small subset of an input graph allows generating realistic graphs that can be used in the context of algorithm benchmarks or simulations.

After designing and implementing both algorithms, their performance was evaluated in a test framework on several real social network graphs with attributes. The evaluation helped to quantify the influence of algorithm parameters on their performance and to find optimum values for these parameters. Comparisons indicate that the developed mechanisms are on a par with state-of-the-art algorithms when it comes to performance with respect to topological aspects. However, the implemented algorithms come out ahead when graphs with node attributes are to be analyzed or generated. Further experiments show that small sample sizes are sufficient for reliable estimates of topological and attribute related graph properties. Therefore, the developed algorithms provide time and resource efficient means of analysis and generation of graphs with node attributes.

In the context of socially aware traffic management, for example, sampling methods that take into account not only topology but also graphs' attribute values can contribute to enhancing existing traffic management techniques with social information. Time and resource efficient analysis of social network data (e.g., users' interests) allows ISPs to employ techniques like prefetching and caching in order to minimize expenses for inter-AS traffic and improve QoE for end users. Graph generation algorithms, on the other hand, can help researchers conduct performance evaluation of graph algorithms by providing realistic input data.

The performance evaluation of the algorithms focused on social network graphs, which are highly relevant because of their size and the usability of obtained insights. An open point remains to what extend the proposed algorithms can be applied to other graphs than social networks. As the design of the algorithms is very general, a decent performance for all types of networks can be expected. However, an extensive evaluation of the sampling and generation algorithms for a huge variety of graphs remains for future work.

With many other possible applications in a variety of use cases, the developed algorithms provide a solid foundation for further research. In the future, research might focus on extending the graph generation algorithm with scaling capabilities. This would allow generating similar graphs of arbitrary sizes from a single input graph. Thus, phenomena like the temporary evolution of OSN graphs could be simulated. Additionally, mechanisms for estimating missing attribute values in user profiles and practical extensions to multiple attributes would allow creating even more accurate and thus realistic output graphs. Finally, extending the generation algorithm's capabilities so that it can handle larger networks can increase the range of possible applications.

Acknowledgments

This work was partly funded in the framework of the EU ICT Project SmartenIT (FP7-2012-ICT-317846). The authors alone are responsible for the content.

References

[1] N.K. Ahmed, J. Neville, R. Kompella, Network Sampling: From Static to Streaming Graphs 8 (2) (2012) 7 arXiv:1211.3412v1.
[2] M. Berlingerio, D. Koutra, T. Eliassi-Rad, C. Faloutsos, Network similarity via multiple social theories, in: Proceedings of IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, 2013.
[3] P. Bonacich, Factoring and weighting approaches to status scores and clique identification, J. Math. Sociol. 2 (1) (1972) 113–120.
[4] G. Daraganova, P. Pattison, Exponential Random Graph Models for Social Networks, Cambridge University Press, 2012.
[5] X. Dimitropoulos, D. Krioukov, A. Vahdat, G. Riley, Graph annotations in modeling complex network topologies, ACM Trans. Model. Comput. Simul. (TOMACS) 19 (4) (2009) 17.
[6] P. Fronczak, A. Fronczak, M. Bujok, Exponential random graph models for networks with community structure, Comput. Res. Repository 88 (3) (2013).
[7] M. Gjoka, M. Kurant, C.T. Butts, A. Markopoulou, Walking in Facebook: a case study of unbiased sampling of OSNs, in: Proceeddings of IEEE INFOCOM, 2010.
[8] M. Gjoka, M. Kurant, A. Markopoulou, 2.5K-Graphs: from sampling to generation, in: IEEE INFOCOM, 2013.
[9] M. Gjoka, B. Tillman, A. Markopoulou, Construction of simple graphs with a target joint degree matrix and beyond, in: IEEE INFOCOM, 2015.
[10] M.H. Hansen, W.N. Hurwitz, On the theory of sampling from finite populations, Annal. Math. Stat. 14 (4) (1943) 333–362.
[11] M.R. Henzinger, A. Heydon, M. Mitzenmacher, M. Najork, On near-uniform URL sampling, Comput. Netw. 33 (1) (2000) 295–308.
[12] J. Illenberger, G. Flötteröd, Estimating properties from snowball sampled networks, Technical Report, TU Berlin, Transport Systems Planning and Transport Telematics, 2011.

[6] Intel Core i7 4770 CPU at 3.40 GHz with 16 GB of RAM.

[13] L. Katzir, E. Liberty, O. Somekh, I.A. Cosma, Estimating sizes of social networks via biased sampling, Internet Math. 10 (3–4) (2014) 335–359.

[14] M. Kim, J. Leskovec, Modeling social networks with node attributes using the multiplicative attribute graph model, in: Proceedings of the Conference on Uncertainty in Artificial Intelligence, 2011.

[15] M. Kim, J. Leskovec, Multiplicative attribute graph model of real-world networks, Internet Math. 8 (1–2) (2012) 113–160.

[16] J. Koskinen, D. Lusher, G. Robbins, Exponential Random Graph Models for Social Networks: Theory, Methods, and Applications, Cambridge University Press, 2013.

[17] M. Kurant, A. Markopoulou, P. Thiran, On the bias of BFS (Breadth First Search), in: International Teletraffic Congress, 2010.

[18] M. Kurant, A. Markopoulou, P. Thiran, Towards unbiased BFS sampling, Comput. Res. Repository 29 (9) (2011) 1799–1809.

[19] J. Leskovec, C. Faloutsos, Sampling from large graphs, in: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2006.

[20] L. Lovász, Random walks on graphs: a survey, Combinatorics 2 (1) (1993) 1–46.

[21] P. Mahadevan, C. Hubble, D. Krioukov, B. Huffaker, A. Vahdat, Orbis: Rescaling degree correlations to generate annotated Internet topologies, in: Proceedings of ACM SIGCOMM Computer Communication Review, 2007.

[22] P. Mahadevan, D. Krioukov, K. Fall, A. Vahdat, Systematic topology analysis and generation using degree correlations, Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement (2006).

[23] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, E. Teller, Equation of state calculations by fast computing machines, J. Chem. Phys. 21 (6) (1953) 1087–1092.

[24] A. Mislove, M. Marcon, K.P. Gummadi, P. Druschel, B. Bhattacharjee, Measurement and analysis of online social networks, in: Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement, 2007.

[25] M. Najork, J.L. Wiener, Breadth-first crawling yields high-quality pages, in: Proceedings of the 10th International Conference on World Wide Web, 2001.

[26] A.H. Rasti, M. Torkjazi, R. Rejaie, N. Duffield, W. Willinger, D. Stutzbach, Respondent-driven sampling for characterizing unstructured overlays, in: Proceedings of IEEE INFOCOM, 2009.

[27] B. Ribeiro, D. Towsley, Estimating and sampling graphs with multidimensional random walks, in: Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement, 2010.

[28] G. Robins, P. Elliott, P. Pattison, Network models for social selection processes, Soc. Netw. 23 (1) (2001) 1–30.

[29] M. Seufert, G. Darzanos, I. Papafili, R. Lapacz, V. Burger, T. Hoßfeld, Socially-aware traffic management, in: K.A. Zweig, W. Neuser, V. Pipek, M. Rohde, I. Scholtes (Eds.), Social Informatics – The Social Impact of Interactions between Humans and IT, Springer, 2014.

[30] T.A.B. Snijders, P.E. Pattison, G.L. Robins, M.S. Handcock, New specifications for exponential random graph models, Sociol. Method. 3 (2) (2006) 1–40.

[31] D. Stutzbach, R. Rejaie, N. Duffield, S. Sen, W. Willinger, On unbiased sampling for unstructured peer-to-peer networks, IEEE/ACM Trans. Network. 17 (2) (2009) 377–390.

[32] L. Takac, M. Zabovsky, Data analysis in public social networks, in: Proceedings of the International Scientific Conference and International Workshop Present Day Trends of Innovations, 2012.

[33] A.L. Traud, P.J. Mucha, M.A. Porter, Social structure of Facebook networks, Phys. A: Stat. Mech. Appl. 391 (16) (2012) 4165–4180.

[34] T. Wang, Y. Chen, Z. Zhang, P. Sun, B. Deng, X. Li, Unbiased sampling in directed social graph, in: Proceedings of ACM SIGCOMM Computer Communication Review, 2010.

[35] S. Wasserman, P. Pattison, Logit models and logistic regressions for social networks: I. an introduction to markov graphs and $p*$, Psychometrika 61 (3) (1996) 401–425.

[36] H. Yun, S.V.N. Vishwanathan, Efficiently sampling multiplicative attribute graphs using a ball-dropping process, Comput. Res. Repository abs/1202.6001 (2012).