

Reconsidering the Foundations of Network Sampling

Nesreen K. Ahmed, Jennifer Neville, Ramana Kompella
Purdue University, Department of Computer Science
{nkahmed,neville,kompella}@cs.purdue.edu

1. Introduction

Recently, there has been a great deal of research focusing on the development of sampling algorithms for networks with small-world and/or power-law structure. The peer-to-peer research community (e.g., [7]) have used sampling to quickly explore and obtain a good representative sample of the network topology, as these networks are hard to explore completely and have significant amounts of churn in their topology. For collecting data from social networks, researchers often use snowball sampling (e.g., [2]) due to the lack of access to the complete graph. Leskovec et al. have developed Forest Fire Sampling, which uses a hybrid combination of snowball sampling and random-walk sampling to produce samples that match the temporal evolution of the underlying social network [5]. Hubler et al. have developed a Metropolis algorithm which samples in a manner designed to match desired properties in the original network [3].

Although there has been a great deal of research focusing on the the development of sampling *algorithms*, much of this work is based on empirical study and evaluation (i.e., measuring the similarity between sampled and original network properties). There has been some work (e.g., [4, 8, 6]) that has studied the statistical properties of samples of complex networks produced by traditional sampling algorithms such as node sampling, edge sampling and random walks. However, there has been relatively little attention paid to the development of a theoretical foundation for sampling from networks—including a formal framework for sampling, an understanding of various network characteristics and their dependencies, and an analysis of their impact on the accuracy of sampling algorithms. In this paper, we reconsider the *foundations* of network sampling and attempt to formalize the goals, and process of, sampling, in order to frame future development and analysis of sampling algorithms.

2. Network Sampling

We start by defining *population* as a set of representative units of interest. The term *sampling* refers to the process of selecting a subset of the population, in order to measure certain characteristics of the subset, and then use the measurements to estimate properties of the whole population.

Much of the previous research on graph and network sampling has focused on algorithm development (i.e., sampling method). For example, research has focused on developing algorithms to sample small(er) graphs from a single large graph. There are been little attention paid to the other dimensions of the sampling process. For example, to the best of our knowledge, none of the recent work on network sampling includes an explicit definition of the population of interest or a description of the set of units under consideration. In addition, the proposed algorithms are typically evaluated by measuring the similarity between characteristics of sampled graphs and the original graph. This type of evaluation is an indirect *proxy* for “representativeness” and is not necessarily an accurate measure of sampling error.

Below, we explicitly consider the aims and decisions of the sampling process in the context of networks. The goal is to provide a solid methodological foundation for network sampling, from which we can empirically compare various graph sampling algorithms, and understand their relative strengths and weaknesses. Specifically, we outline and discuss the following four steps of the network of the sampling process: (1) define the **study objectives**, (2) define the **elementary units** and **population**, (3) choose a **sampling procedure**, and (4) evaluate the **sampling error** in either the parameter estimates or the representativeness of the samples. We discuss each of these in more detail next.

2.1. Study Objectives

A *statistical population* is typically defined as the set of all items that one wishes to study. In network sampling, the explicitly-stated goal is often to select a smaller graph from a larger network. However, there are often two different objectives of network sampling that are not explicitly stated:

1. To accurately estimate global network parameters (e.g., density, degree).
2. To select a subgraph with representative graph structure (e.g., to evaluate classification algorithms or network protocols overlaid on the graph structure).

Note that these objectives both focus on characteristics of entire networks rather than nodes or edges. In

addition, these two objectives may be difficult to satisfy simultaneously—i.e., if the sampled data enable accurate study of one, it may not allow accurate study of the other. We will discuss this issue in more detail in Section 2.4.

2.2. Units and Population

Once the study objectives are outlined, the population of interest can be defined by extension. In many cases, the definition of the population may be obvious. For example, if one wishes to study the characteristics of deciduous trees in the Midwest, the population is explicitly defined by the goal of the study. The main challenge is then to select a representative subset of trees (i.e., units) from this population in order to make the study cost efficient and feasible.

Other times, the population may be less tangible and difficult to define. For example, if one wishes to study the characteristics of a system or *process*, there is not a clearly defined set of items to study. Instead, one is often interested in the overall behavior of the system. In this case, the population can be defined as the set of possible outcomes from the system (e.g., measurements over all settings) and these *units* should be sampled according to their underlying probability distribution.

In the two network sampling objectives outlined above, the object of study is an entire network (either for structure or parameter estimation). This means that in these cases, the elementary units should correspond to networks, rather than nodes or dyads (i.e., pairs of linked nodes). As such, the population should be defined as a set of networks of a specified size, or the set of networks that could be generated by the same underlying process that created the input network. In practice, we can rarely observe multiple networks from the same domain. There is only one Internet, one World Wide Web, one Facebook friendship graph—although we can down-sample many smaller networks from these large networks, we cannot measure a second, independent instance. These networks are all *complex systems* evolving over time. Therefore, it may be more reasonable to define the population through the process that underlies the formation of the networks. Although it is difficult to model generative *processes* for networks, one of the primary goals of network analysis is to understand these processes—either the process which drives the network formation or the processes by which activity and information spread throughout the networks. As such, we argue that network processes are generally the true focus of study and populations should be defined accordingly.

The definition of a network population as the set of networks that could be generated from the same underlying process, clarifies several issues in graph sampling—by making the population definition more precise and tying it to study objectives. Figure 1 shows an example of graphs generated by a process \mathcal{P} in the range $n = 2$ to $n = 10,000$.

Let \mathcal{P} be the process which can generate a set of graphs \mathcal{G} (of any size). As shown in figure 1, the possible graphs are represented by a sequence of layers where each layer represents all graphs of a particular size n (ranging from an empty graph to a complete graph). For any two consecutive layers i and j , there are several ways to go from one graph on i to another graph on j by adding only one node and some edges (zero or more). This implies that there is a lattice of all possible graphs with several paths to go from one layer to the next. However, it is not reasonable to assume that every graph is equally likely to be generated by a process \mathcal{P} , since \mathcal{P} specifies the manner in which nodes connect in the graph. Therefore, there should be a probability distribution over graph structures, which specifies a nonnegative probability for the likelihood that a particular graph will be generated by \mathcal{P} . More specifically, we can consider a probability distribution over graphs of a particular size (i.e., n nodes), given the underlying process \mathcal{P} as: $P(G_n|\mathcal{P})$. We assume our input graph G^* of size n is drawn from this distribution. Then if the aim is to select subgraphs of size $m < n$ with representative structure, the sampling algorithm should produce samples according to their likelihood in the distribution $P(G_m|\mathcal{P})$. Thus algorithm evaluation could assess sample “representativeness” by estimating an empirical distribution for $\hat{P}(G_m|\mathcal{P})$ from the generated samples and compare it to $P(G_m|\mathcal{P})$. If, on the other hand, the aim is to estimate population parameters for graphs of size n , then the sampling algorithm should select networks of size m , such that functions of those networks are equal to those of the larger graphs, i.e., $\mathcal{F}(G_m^s) = \mathcal{F}(G_n)$, where $\mathcal{F}(g)$ is for example, the density of the graph g .

One advantage of this view is that it makes the role of graph size explicit. If the aim is to obtain a sample of 10% of the nodes from the larger input graph that accurately reflects the graph structure of the underlying process \mathcal{P} , then sampling algorithms should produce sample graphs with a level of connectivity that would reasonably be generated from \mathcal{P} with a smaller set of nodes. Indeed, this points to a flaw in current evaluation methods, which typically match metrics on the (smaller) sample graph to those observed on the (larger) original graph without a clearly defined sampling objective. In practice, however, our knowledge of the underlying process \mathcal{P} will be limited. In particular, we typically have only a single (albeit large) network drawn from \mathcal{P} , which makes it difficult to infer $P(G_m|\mathcal{P})$.

2.3. Example Sampling Methods

Once the population has been defined, a sampling method must be chosen to select units from the population. Here we describe three example sampling algorithms for networks: node sampling, forest fire sampling [5], and streaming time node sampling [1].

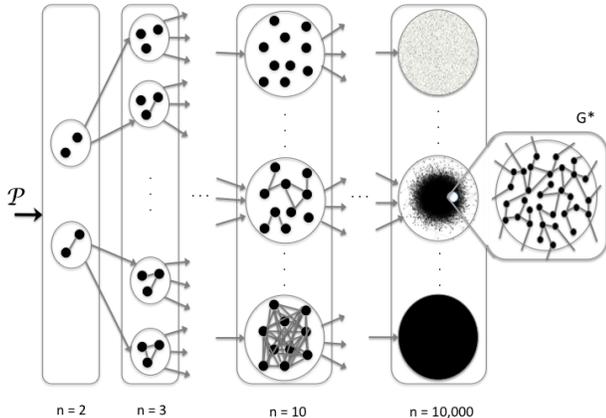


Figure 1. Example graph generation process.

Node sampling In node sampling (NS), nodes are chosen independently and uniformly at random from the original graph for inclusion in the sampled graph. Once the sample reaches the desired size, all edges among the sampled nodes in the original graph are selected for inclusion in the sample.

Forest Fire Sampling Forest Fire sampling (FFS) [5] is a combination of node and edge sampling—nodes are first selected based on a random walk through the graph, then all edges among the selected nodes are chosen for the sample. The FFS algorithm starts by picking a node uniformly at random and adding it to the sample. Then the algorithm ‘burns’ a fraction of its outgoing links with the nodes attached to them. The fraction is a random number drawn from a geometric distribution with mean $p_f/(1-p_f)$. (For the experiments in this paper, we use $p_f = 0.7$ as recommended in [5]). This process is recursively repeated for each neighbor that is burned until no new node is selected to be burned. If that occurs, a new node is chosen at random from the graph to start the process again. The process is continued until the desired number of nodes is selected.

Streaming Time Node Sampling Streaming Time Node Sampling (STNS) [1] is also a combination of node and edge sampling, where nodes are selected proportionally based on their communication activity over time. STNS samples all edges within a set of randomly selected time windows and then populates the node set with the corresponding nodes. Among the selected node set, STNS accumulates additional edges that occur due to communication outside the sampled time windows. The algorithm starts by scanning the activity timeline in a streaming fashion. A timestamp t is selected from the activity timeline according to a Bernoulli process with mean $p_s = m/T$, where T is the total number of timestamps and m is the average number of timestamps needed to sample the desired number of nodes. STNS selects the nodes incident on activity edges that occur in the next time window $([t, t + \tau])$, in this work we use $\tau = 1$ day). The process is repeated in a streaming fashion

until the appropriate fraction of nodes is collected. Each node v is associated with a sample time st_v for when it is added to the sample. Then the algorithm chooses the edges $e = (v_i, v_j, t)$ such that $t \geq st_{v_i}$ and $t \geq st_{v_j}$. In other words, STNS only samples an activity edge that involves a particular node if it occurred *after* that node has been added to the sample.

Note that these algorithms focus only on obtaining some (random) selection of nodes or edges, as the units implicitly considered by these algorithms are either nodes or edges. To the best of our knowledge, there are no sampling algorithms that explicitly consider a population in which the elementary units are ‘networks’. While designing such an algorithm is still an open question, in the meantime, we investigate how the performance of existing algorithms compares when considering the objectives from Section 2.1.

2.4. Evaluating Sampled Networks

The evaluation of sampling algorithm accuracy should be tied to the study objectives—either measuring the quality of parameter estimates inferred from the samples, or measuring the representativeness of the sampled graph structure. However, networks sampling accuracy is often evaluated by comparing graph metrics (e.g., degree distributions) on the original and (smaller) sampled graphs. This approach makes at least two implicit assumptions about the evaluation procedure.

First, for the aim of selecting *representative* subnetworks, similarity in graph statistics only indirectly represents the underlying likelihood of the sampled network. That is, instead of measuring the likelihood of the subnetwork given the underlying process that generated the input graph directly, researchers consider the input graph statistics to be the target and measure the deviation of the subnetwork statistics from this target. Since this evaluation is substituted for the likelihood, we call it a proxy evaluation. The primary assumption with this proxy evaluation is that when the sampled network exhibits graph metrics similar to the original input network, then the sample is “close” to the mode of the distribution. Since the underlying distribution is not formally defined, it is not clear whether this assumption holds in practice. Moreover, when the statistics do not match exactly, the implicit assumption is that “closer” implies “more” representative. Again, it is not clear whether this holds for real world network processes. Furthermore, this indirect evaluation leaves open the possibility that other graph metrics might be better proxies for likelihood.

Second, for the aim of inferring *population* parameters of the original graph, using statistics from a smaller sampled graph implicitly assumes that the graph metrics are *independent* of graph size. However, this is unlikely to be true for many commonly used metrics. For example, consider the case where the original network consists of n_o nodes and

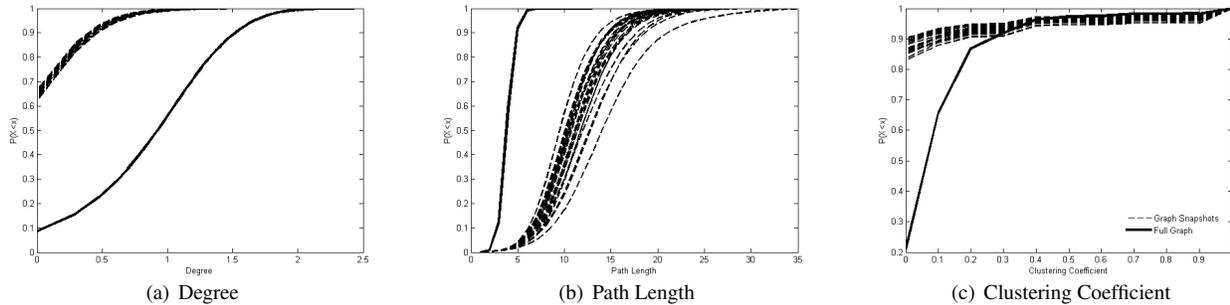


Figure 2. Cumulative distributions of graph metrics for full and reduced size graph snapshots .

we construct a 10% sample (i.e., $n_s = 0.1n_o$). Let $|E_o|$ and $|E_s|$ be the number of edges in the original and the sampled network respectively, and let the density in the original network be $den_o = \frac{|E_o|}{n_o(n_o-1)}$. Then if we match the density in the sampled graph: $den_s = \frac{|E_s|}{0.1n_o(0.1n_o-1)} = \frac{|E_o|}{n_o(n_o-1)}$, the number of sample edges will be: $|E_s| \simeq 0.1^2|E_o| < 0.1|E_o|$. This shows the dependency of graph metrics on graph size. On the original graph, the average degree is $\bar{d}_o = \frac{|E_o|}{n_o}$. On a smaller sample graph, if the density is equal to that of the original graph, then the average degree will be underestimated $\bar{d}_s = \frac{|E_s|}{0.1n_o} \simeq \frac{0.1|E_o|}{n_o}$. Similarly, if we aim to capture the original average degree in the sample, then the density will be overestimated. It is not clear which metric to optimize to produce “better” sample graphs.

The above argument demonstrates the connection between graph size and global network parameters theoretically. We can also investigate this empirically on real social network data generated by a process over time. Specifically we consider on a *social activity* network that consists of ‘mini’ communications among the set of 56,061 publicly visible Facebook users in a university network. Each Facebook user has a public message board called the ‘Wall’ on which other users can post small messages. From the Wall postings that span a whole year period, we construct a *Wall graph* with links from senders to different receivers. Each link is associated with the timestamp of the communication.

In figure 2, we show cumulative distributions for several network parameters, for both the original graph (i.e., $n = 56061$) and a set of 25 smaller networks generated by the same underlying process. Since the original graph was constructed by aggregating social network activity over the span of a year, we can construct smaller networks from the same process by aggregating over windows of less than one year. Each small network is 20% of the size of the full graph (i.e., $m = 0.20n$). To collect a small network, we start from a randomly selected timestamp and add all edges with their incident nodes that appear after the timestamp, continuing the process until we reach the desired network size.

Figures 2(a), 2(b), and 2(c) show the degree, path length, and clustering coefficient distributions respectively for the

full graph and a set of 25 temporal graph snapshots sampled from the underlying process. In the figures it is clear that the graph distributions, for the original network and the set of smaller networks, do not overlap and moreover, they show a distinct difference for the two different graph sizes. Specifically, the distributions for the small networks all have a higher fraction of low degree nodes, longer path lengths, and less clustering. This illustrates other potential dependencies between graph metrics and graph size and is thus more evidence that sampling algorithms will not be able to simultaneously satisfy objectives 1 and 2. We note however, that here we assume a stationary generative process for the networks. However, it’s possible that non-stationarity in the real data may impact the observed difference in graph metrics for different sized networks.

In the next section, we present an initial exploration of how the three example sampling algorithms perform for two different sampling objectives. Specifically, we use temporal activity graphs and measure network characteristics as they evolve over time in order to compare statistics to those of the original graph and the original graph when it was a smaller size.

3. Empirical Evaluation of Network Sampling

In the experimental evaluation, we compare STNS [1] to NS, and FFS [5]. In this comparison, we aim to explore how the three algorithms perform for the two different sampling objectives. We select these three algorithms since they have been shown to perform relatively well in several problems. Node sampling captures degree distributions due to its inclusion of all edges for a chosen node set, however, the original connectivity is less likely to be preserved [4]. Thus, NS can lead to biased estimates of clustering and path length properties. In some other situations, however, researchers have observed that NS produces reasonably good samples [5]. On the other hand, forest fire sampling has been shown to produce quite accurate samples in practice that match the original graph’s degree distribution and path length distribution. In addition, for temporal activity networks, STNS has been shown to capture not only degree and path lengths, but also clustering quite well [1]. We

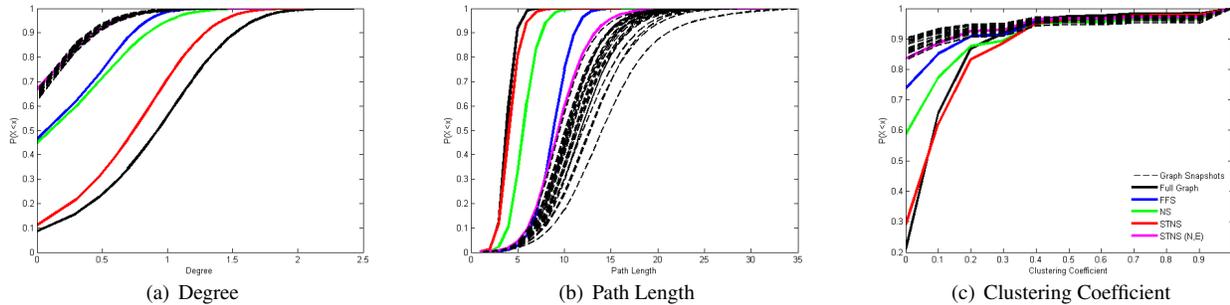


Figure 3. Cumulative distributions of graph metrics for original, reduced size, and sampled networks.

also compare to a variant of STNS, which we refer to as STNS(N,E), which has two sampling parameters, one for the target number of nodes and another for the target number of edges. In STNS(N,E), we constrain the number of edges to be the number of edges in the graph when its size equals to the sample size back in time, only sampling edges until we meet this constraint.

We evaluate the four sampling algorithms on the Facebook data, comparing against both the full graph and against the set of 25 smaller-sized networks. Figure 3(a), 3(b), and 3(c) show the degree, path length, and clustering coefficient distributions respectively for the three sampling algorithms.

From the figures, we can make the following observations. First, the distributions of the sampling algorithms are different even they have the same number of nodes (20%). STNS preserves all the distributions of the full graph better than both NS and FFS, therefore STNS satisfies objective 1. However, none of the three algorithms produce a sample that accurately captures the graph structure at the smaller size (i.e., 20% n). Therefore, none of the three algorithms satisfy objective 2. Both FFS and NS fall in the middle between the distributions of the two graph sizes. This implies that FFS and NS neither satisfy objective 1 nor objective 2.

The differences between NS, FFS, and STNS is due to which nodes are added to the sample (low degree versus high degree nodes) and also how many edges are added. In general, STNS captures more edges of the sampled nodes which leads to a skewed sampled network that approximates the statistics of the full graph (objective 1) but does not represent the (smaller) graph structure well (objective 2). However, FFS and NS tend to sample fewer edges, which may be appropriate for smaller graphs, but results in misestimation of the graph metrics for larger graphs (objective 1). However, they are still far from the distributions for the small networks, thus, they fail to satisfy objective 2 as well.

Second, although STNS samples doesn't represent the (smaller) graph structure well, STNS(N,E) produces samples that closely matches the distributions of the 20% networks—in fact for degree and clustering the STNS(N,E) distributions are hard to see because they fall in the same range as the 25 smaller networks. Most existing algorithms

work with only one sampling parameter either for the number of nodes (e.g. node sampling) or for the number of edges (e.g. edge sampling). The STNS(N,E) results indicate that multiple parameters could be used to choose between the two sampling objectives with a single algorithm. However, it is unclear how to set the parameter E if there is no access to the temporal evolution of the network.

4. Conclusion

In this work, we conjecture that a lack of precise problem formulation has made it difficult to accurately compare network sampling algorithms. Poorly specified objectives, lack of population definition, and indirect measurement of sampling error, all make it difficult to evaluate network sampling algorithms. To illustrate this, we showed empirically that current sampling methods perform differently with respect to two different study objectives.

References

- [1] N. K. Ahmed, F. Berchmans, J. Neville, and R. Kompella. Time-based sampling of social network activity graphs. In *Proceedings of the 8th Workshop on Mining and Learning with Graphs*, 2010.
- [2] Y. Ahn, S. Han, H. Kwak, S. Moon, and H. Jeong. Analysis of topological characteristics of huge online social networking services. In *WWW*, pages 835–844, 2007.
- [3] C. Hubler, H.-P. Kriegel, K. M. Borgwardt, and Z. Ghahramani. Metropolis algorithms for representative subgraph sampling. In *ICDM*, 2008.
- [4] S. Lee, P. Kim, and H. Jeong. Statistical properties of sampled networks. *Physical Review E*, 73:016102, 2006.
- [5] J. Leskovec and C. Faloutsos. Sampling from large graphs. In *SIGKDD*, pages 631–636, 2006.
- [6] M. Stumpf, C. Wiuf, and R. May. Subnets of scale-free networks are not scale-free: Sampling properties of networks. *Proceedings of the National Academy of Sciences*, 102:4221–4224, 2005.
- [7] D. Stutzbach, R. Rejaie, N. Duffield, S. Sen, and W. Willinger. On unbiased sampling for unstructured peer-to-peer networks. In *IMC*, pages 27–40, 2006.
- [8] S. Yoon, S. Lee, S.-H. Yook, and Y. Kim. Statistical properties of sampled networks by random walks. *Phys. Rev. E*, 75(4):046114, Apr 2007.