Sampling Techniques for Large, Dynamic Graphs

Daniel Stutzbach, Reza Rejaie
University of Oregon
{agthorr,reza}@cs.uoregon.edu

Nick Duffield, Subhabrata Sen, Walter Willinger
AT&T Labs—Research
{duffield,sen,walter}@research.att.com

***Abstract*— Peer-to-peer systems are becoming increasingly popular, with millions of simultaneous users and a wide range of applications. Understanding existing systems and devising new peer-to-peer techniques relies on access to representative models derived from empirical observations. Due to the large and dynamic nature of these systems, directly capturing global behavior is often impractical. Sampling is a natural approach for learning about these systems, and most previous studies rely on it to collect data.**

This paper addresses the common problem of selecting representative samples of peer properties such as peer degree, link bandwidth, or the number of files shared. A good sampling technique will select any of the peers present with equal probability. However, common sampling techniques introduce bias in two ways. First, the dynamic nature of peers can bias results towards short-lived peers, much as naively sampling flows in a router can lead to bias towards short-lived flows. Second, the heterogeneous overlay topology can lead to bias towards high-degree peers. We present preliminary evidence suggesting that applying a degree-correction method to random walk-based peer selection leads to unbiased sampling, at the expense of a loss of efficiency.

I. Introduction

Peer-to-peer (P2P) systems are becoming increasingly popular, with millions of simultaneous users [1] and covering a wide range of applications, from file-sharing programs like LimeWire and eMule to Internet telephony services such as Skype. Understanding existing systems and devising new P2P techniques relies on having access to representative models derived from empirical observations of existing systems. However, due to the large and dynamic nature of P2P systems, it is often difficult or impossible to directly capture global behavior. Sampling is a natural approach for learning about these systems using light-weight data collection, relied on by most previous studies (*e.g.*, [4], [19]). One challenge, however, is ensuring that the samples are representative (or *unbiased*).

This paper addresses the common problem of selecting representative samples of *peer properties* such as peer degree, link bandwidth, or the number of files shared [24]. To examine peer properties, any sampling technique needs to locate a set of peers in the overlay and gather data from them. Initially, the sampling program is aware of a handful of peers and leveraging them to learn about additional peers. Typically, the sampling program queries known peers to learn about their neighbors, incrementally exploring a fraction of the overlay graph.[1] A good sampling technique will select any of the peers present with equal probability. However, as we will show, commonly used sampling techniques can easily introduce significant bias in two ways. The first cause of bias is the highly dynamic nature of these systems. It is easy to imagine the overlay as a static graph from which we want to collect a set of peers. However, gathering a set of samples takes time, and during that time the graph will change. In Section II-A, we show how this often leads to bias towards short-lived peers and explain how to overcome this difficulty.

The second significant cause of bias is the graph properties of the P2P topology. A naive approach will be heavily biased towards high-degree peers. As the sampling program explores the graph, each link it traverses is much more likely to lead to a high-degree peer than a low-degree peer. We describe different techniques for traversing the overlay to select peers in Section II-B and evaluate them in Section III via simulation. In this preliminary work, we simulate using two types of graphs: ordinary random graphs and an actual snapshot of the Gnutella graph topology [22]. In our ongoing work, we are adding other types of random graphs, such as certain power-law random graphs and small-world graphs, to explore the robustness of the considered techniques to different types of graph structures. By comparing and contrasting the performance of different techniques in different settings, we can gain a better understanding of the most efficient techniques to consistently yield unbiased (or only slightly biased) samples.

In summary, bias in sampling from P2P systems can be introduced along two axes: *(i)* temporal (due to differences in peer lifetimes) and *(ii)* topological (due to differences in peer degree). Our findings show that these factors cause heavy bias in commonly used techniques such as breadth-first search and random walks. We present preliminary evidence suggesting that applying a degree-correction method to random walk leads to unbiased sampling, at the expense of a loss of efficiency. Section IV discusses related work, and Section V concludes the paper with a summary of our findings and plans for future work.

II. Sampling Peer Properties

Our goal in this paper is to tackle the common problem of sampling *peer properties*, which covers a wide range of interesting aspects. Examples include products of user behavior (such as the number of files shared and link bandwidth), local graph properties (such as degree and clustering coefficient), and dynamic properties (such as remaining uptime). Global properties, such as the graph diameter, cannot be determined easily using sampling and tend to rely on heavy-weight solutions, such as crawling the entire overlay [21].

Collecting a sample of a property is a two-step process.

[1] Other sampling programs rely on passive monitoring or querying for popular files, but such approaches are fundamentally biased towards peers generating more traffic or with those files. We do not consider them further.

First, the selection process explores part of the P2P overlay and selects a peer. Second, a property-specific measurement tool gathers the sample. For example, sampling the clustering coefficient requires gathering the neighbor information for the selected peer and all of its neighbors. Sampling the remaining uptime requires monitoring the peer until it departs the network. This paper is concerned with the first step, selecting a peer, which is the common aspect for sampling any peer property.

The goal is to select an *unbiased* sample, meaning selecting the sample uniformly at random. Additionally, the sampling process should also be *efficient*, meaning that the sampling process should not have to explore a large portion of the graph to select an unbiased sample. As we described in Section I, bias can be caused by the dynamic nature of P2P systems and by their graph structure. In the following two sections we introduce mechanisms to cope with these problems.

A. Coping with dynamics

We develop a formal and general model of a P2P system as follows. If we take an instantaneous snapshot of the system at time t, we can view the overlay as a graph $G(V, E)$ with the peers as vertices and connections between the peers as edges. Extending this notion, we incorporate the dynamic aspect by viewing the system as an infinite series of time-indexed graphs, $G_t = G(V_t, E_t)$. The most common approach for sampling from this series of graphs is to define a measurement window, $[t_0, t_0 + \Delta]$, and select peers uniformly at random from the set:

$$V_{t_0,t_0+\Delta} = \bigcup_{t=t_0}^{t_0+\Delta} V_t$$

This formulation is appropriate if peer session lengths are exponentially distributed (*i.e.*, memoryless). However, existing measurement studies [10], [17], [19], [22] show session lengths are heavily skewed, with many peers being present for just a short time (a few minutes) while other peers remain in the system for a very long time (*i.e.*, longer than Δ). As a consequence, as Δ increases, the set $V_{t_0,t_0+\Delta}$ includes an increasingly large fraction of short-lived peers.

A simple example may be illustrative. Suppose we wish to observe the number of files shared by peers. In this example system, half the peers are up all the time and have many files, while the other peers remain for around 1 minute and are immediately replaced by new short-lived peers, who have few files. The technique used by most studies would observe the system for a long time (Δ) and incorrectly conclude that most of the peers in the system have very few files. Moreover, their results will depend on how long they observe the system. The longer they watch, the larger the fraction of observed peers with few files.

One fundamental problem of this approach is that it focuses on sampling *peers* instead of *peer properties*. It selects each sampled vertex at most once. However, the property at the vertex may change with time. Our goal should not be to select a vertex $v_i \in \bigcup_{t=t_0}^{t_0+\Delta} V_t$, but rather to sample the property at v_i at a particular instant t. This means we must view $v_{i,t}$ and $v_{i,t'}$ as distinct samples even though they come from the same peer. *The key difference is that it must be possible to sample from the same peer more than once, at different points in time.* We may accomplish this goal by sampling selecting t and $v_{i,t}$ uniformly from the sets:

$$t \in [t, t_0 + \Delta],\ v_{i,t} \in V_t$$

This sampling technique will not be biased by the dynamics of peer behavior, because the sample set is decoupled from peer session lengths. To our knowledge, no prior P2P measurement studies relying on sampling use this approach.

Returning to our simple example, this approach will correctly select long-lived peers half the time and short-lived peers half the time. When the samples are examined, they will show that half of the peers in the system at any given moment have many files while half of the peers have few files, which is exactly correct.

We can now divide the sampling process into two parts: *(i)* selecting times uniformly at random and *(ii)* selecting peers uniformly at random from all peers available at that time. Selecting times uniformly at random can be easily achieved by generating times between samples using an exponential distribution. At each chosen time, we must collect a sample from the peers present at that time, which reduces to the problem of selecting a vertex uniformly at random from a graph. We address this problem in the next subsection.

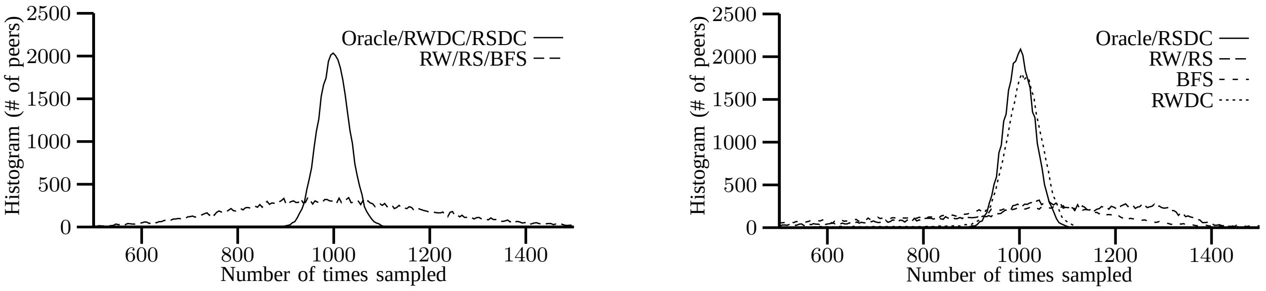
B. Coping with graph structure

In this section, we discuss several techniques for selecting vertices randomly from a graph. When sampling from a P2P system, we typically begin with knowledge of at least one peer and a method to query known peers for a list of their neighbors. The goal is to explore a small fraction of the graph yet return a peer (vertex) uniformly at random. In Section III, we will evaluate the techniques discussed below using simulation.

Two classical ways to explore a graph are via breadth-first (BFS) and depth-first search (DFS), often used by sampling techniques that crawl a portion of the overlay topology (as in [19]). These techniques add newly discovered peers to a queue and choose new peers to explore by removing them from the queue. They differ only in that BFS uses a FIFO queue while DFS uses a LIFO queue. Neither of these techniques allows duplicates, automatically causing bias towards short-lived peers as described in the previous subsection. We nevertheless include BFS in our simulations, to demonstrate that it performs poorly even in a static system.

Another family of techniques are based on conducting a random walk. The simplest approach is to perform a random walk of length r, select the ending peer as a sample, then perform another walk of length r to get the next sample. While this technique offers low bias for some types of graphs, its efficiency is somewhat low ($\frac{1}{r}$). Graph theory [8], [15] suggests that a good choice is $r \geq \log |V|$.

A more efficient technique performs a random walk of length r, returns that peer as a sample, then continues to



(a) Random Graph

(b) Gnutella

Fig. 1. Bias of different sampling techniques; after collecting $k \cdot |V|$ samples, the figures show how many peers (y-axis) were selected x times

	Oracle	RWDC	RSDC	RW	RS	BFS
Std. Deviation	32	32	32	206	207	210
Skew	0.03	0.03	0.03	0.21	0.21	0.22
Kurtosis	-0.01	-0.01	0.00	0.04	0.03	0.03
Efficiency	100%	2%	4%	8%	99%	99%

(a) Random Graph

	Oracle	RWDC	RSDC	RW	RS	BFS
Std. Deviation	32	65	32	865	866	806
Skew	0.03	-4.28	0.03	47	47.92	17
Kurtosis	-0.01	30	0.00	3084	3087	703
Efficiency	100%	2%	4%	8%	99%	99%

(b) Gnutella

TABLE I

BIAS OF DIFFERENT SAMPLING TECHNIQUES; STATISTICS CORRESPONDING WITH FIGURE 1

walk and return every additional peer along the walk as a sample [8]. However, by not walking r steps between every sample, the samples may be correlated due to the inherit relationship between adjacent peers. We call this technique a "random stroll". This technique is similar to DFS, except it allows duplicates. Since we prefer algorithms that allow duplicates, we omit DFS from our evaluations.

One problem with random walk techniques is that they are biased towards high-degree peers. It is well-known that they visit peers with frequency proportional to the peer's degree [15]. One way to compensate for this problem is to alter the sample-selection criteria slightly. If a peer is a candidate for sampling, select it with probability $\frac{1}{d}$ where d is the peer's degree, otherwise continue the walk and consider the next peer.[2]

For comparison purposes, we can define an ideal sampling technique that uses an oracle to select a peer uniformly at random from all peers that are currently present. While often impractical on real P2P networks, we can easily select peers uniformly at random in a simulator. There is no bias because the selection is not correlated with *any* other peer properties. In summary, we consider the following techniques:

- Uniformly random (Oracle)
- Breadth-first search (BFS)
- Random walk (RW)
- Random stroll (RS)
- Random walk with degree correction (RWDC)
- Random stroll with degree correction (RSDC)

III. Evaluation

In Section II-B we defined several techniques for sampling peers from a P2P system. In this section, we use simulation to explore the performance of these techniques according to three criteria:

- **Bias**: Selecting some peers over others
- **Correlation**: Selecting related peers
- **Efficiency**: How much work is done to collect samples

In this preliminary work, we examine the behavior of sampling techniques over two types of graphs: *(i)* ordinary random graphs and *(ii)* a Gnutella ultrapeer topology snapshot from February 2005, examined in detail in our previous work on characterizing the Gnutella topology [22]. To make useful comparisons, the random graphs have the same number of vertices (161,680) and edges (1,946,596) as the Gnutella topology. To generate edges for the random graphs, we select pairs of nodes at random until we have the desired number of edges, skipping duplicate edges and self-edges.[3] We chose to use these random graphs because they have simple properties and are easy to understand, making them a good baseline for comparisons. We chose the Gnutella topology to examine how the sampling techniques would behave on a real system. Compared to a random graph, the Gnutella topology's degree distribution is significantly more skewed, and it has significantly more clustering. In our ongoing work, we are exploring the robustness of these sampling techniques over a wide variety of common types of graphs.

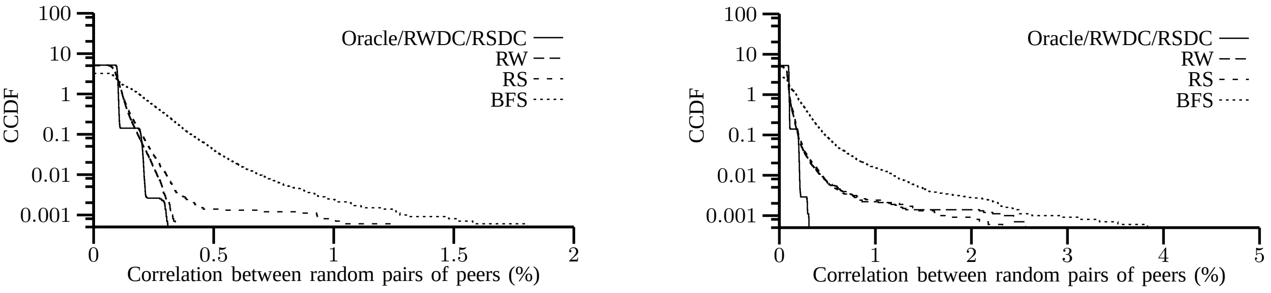
A. Measuring Bias

Uniformly random sampling (*e.g.*, using an oracle) will select each peer with equal probability. A poor sampling technique will select some peers with much greater probability than others. In a simulator, we can compare other sampling techniques to the ideal as follows. For some graph $G(V, E)$, we use each sampling technique[4] to select a very large number of samples, $k \cdot |V|$ (for example, $k = 1000$). We record how many times each node is selected. The typical node should be selected k times, with other nodes being selected

[2] We would like to thank Christos Gkantsidis of Microsoft Research for suggesting this technique.

[3] This process is not guaranteed to generate a connected graph, but will do so with high probability.

[4] Since BFS does not allow duplicates, it cannot sample $k \cdot |V|$ peers in one execution. To simulate realistic usage, we initially perform one random walk to reach a random starting point, then perform a BFS to collect 1,000 samples. We reinitialize the search and repeat until we have $k \cdot |V|$ samples.



(a) Random Graph

(b) Gnutella

Fig. 2. Correlation of different sampling techniques; after collecting $1000 \cdot |V|$ samples, the figures show for a pair of peers (A, B) what percentage of the time (x-axis) did B appear whenever A appeared, as a CCDF over 1 million pairs of (A, B).

close to k times approximately following a normal distribution with variance k.[5] A good sampling technique must produce a similar distribution close to selecting uniformly at random. If the variance is higher, the technique is biased, unfairly selecting some peers more than others.

If a candidate technique produces a distribution similar to the ideal, this is evidence that the technique is unbiased. Although it may be possible to deliberately construct a bad sampling technique that would pass this test, in practice a sampling technique with a *systematic bias* will have significantly more variance than the ideal. Some techniques may not even produce a normal distribution, resulting in high skew and kurtosis (statistics which are very close to zero for samples from a normal distribution).

The results for each of our candidate techniques are shown in Figure 1 using $k = 1000$. Additionally, Table I presents the standard deviation, skew, and kurtosis. In cases where multiple lines were visually indistinguishable, we have plotted only one of the lines for clarity. Specifically, RSDC performs just as well as selecting peers uniformly at random using an oracle. On the other hand, without degree correction both random walk (RW) and random stroll (RS) perform poorly, exhibiting significantly higher standard deviation than the ideal. BFS also exhibits significant bias. The bias of these techniques is also evidenced by large standard deviations, as shown in Table I.

Comparing the data for ordinary random graphs (Fig. 1(a), Tab. I(a)) and the Gnutella topology (Fig. 1(b), Tab. I(b)) several things become apparent. First, we see that Oracle and RSDC perform the same on both types of topologies, evidence that RSDC is unbiased and not adversely affected by graph structure. Second, RWDC performs the same as Oracle and RSDC on the random graph but is slightly skewed on the Gnutella topology. We are unsure what introduces this bias in RWDC, but not RSDC, and plan to study this in our ongoing work by looking for patterns across the over-sampled and under-sampled peers from RWDC. Third, we see that the results for RW, RS, and BFS appear normally distributed for ordinary random graphs but not for the Gnutella topology. In addition to drawing this conclusion based visually on the presence or absence, respectively, of the bell-shaped curve centered around the mean ($k = 1000$), the data in Table I provides further evidence. For random graphs, the skew and kurtosis for these techniques is close to zero, suggesting normality. For Gnutella, the skew and kurtosis are quite large. The bias in these techniques is caused primarily by selecting peers with higher degree, which explains why the results are normally distributed for ordinary random graphs (which have an approximately normally degree distribution) but not for the Gnutella topology (which does not).

Finally, we see that BFS behaves similarly to RW and RS for ordinary random graphs but not for the Gnutella topology. Again, this is a result of the graphs' degree distributions. These techniques respond the same way to normally distributed node degrees, but respond differently to a more skewed distribution. Specifically, the RW and RS techniques are very prone to repeatedly selecting the few high-degree peers in Gnutella. Because BFS maintains a short history and will not select the same high-degree nodes during the same sampling session, it is somewhat more balanced, thus leading to a somewhat lower skew and kurtosis (as shown in Table I(b)).

In summary, BFS, RW, and RS exhibit significant bias. Degree correction for random walk and random stroll cause these techniques to perform well, with RSDC exhibiting no bias on either graph type.

B. Measuring Correlation

A technique that has an equal probability of selecting each peer may still tend to select peers in groups. That is, the results may be *correlated*. BFS is an obvious example of a technique with correlation; if a peer is selected, it becomes very likely its neighbors will also be selected in the same sampling session. Likewise, random stroll may exhibit correlation since it selects neighboring peers.

One method of measuring correlation is to examine the distribution of the percentage of sampling sessions in which node A is selected that also include node B, for all nodes A and B. We define a sampling sessions as a set of 1,000 consecutive samples. A good sampling technique will show a very low percentage for every possible pair. A sampling technique with significant correlation will contain some pairs of peers that frequently appear together. If we plot the distribution as a CCDF, this poor behavior will manifest as a long tail. However, this method requires $O(n^2)$ memory, which is somewhat prohibitive for $n = 161,680$. To overcome this difficulty, we

[5] Based on the normal approximation of a binomial distribution with $p = \frac{1}{|V|}$ and $n = k|V|$

randomly select a large subset (1 million) of the possible pairs of nodes and examine the correlation between only those pairs.

The results are shown in Figure 2. As expected, breadth-first search (BFS) exhibits significantly more correlation (a longer tail) than any of the other techniques, followed by RS. Interestingly, RSDC appears to perform just as well as Oracle. The degree correction causes the random stroll to take extra steps between selections, greatly decreasing the amount of correlation. RWDC also performs well.

Random walk without degree correction performs well over the ordinary random graphs but exhibits slight correlation over the Gnutella topology. This is again a case where the degree distribution affects the performance of the sampling technique. Over the Gnutella topology, the sampling process for RW is so heavily biased (as shown in the previous subsection) by the degree distribution that it causes correlations to occur. In other words, a sampling session often returns a similar set of high-degree peers. In the ordinary random graph, the bias is not strong enough to cause significant correlation since none of the peers are of exceptional degree.

C. Measuring Efficiency

Aside from bias, another important metric for evaluating the usefulness of a sampling technique is its efficiency. One reason for sampling is to reduce the amount of work required to collect useful data. If the sampling technique is inefficient, it does not achieve that goal as well as an efficient technique. Initially, any sampling technique begins with knowledge of a small set of peers in the system. It iteratively queries peers for a list of their neighbors and returns a subset of these discovered peers as the samples. As the basic operation is the neighbors-query, we measure the efficiency as follows:

$$\text{efficiency} = \frac{\text{number of samples produced}}{\text{number of peers queried}}$$

A technique that is 100% efficient returns a sample set containing every peer that it queried. The efficiency does not reveal anything about the quality of the samples; it is simply a measure of how easily the samples are collected.

The efficiencies of the various techniques we examine are shown in the bottom row of Table I. BFS and RS are both very close to 100% efficient. However, as the previous subsections have shown, they are also heavily biased. RW, in addition to being biased, is only 8% efficient. RWDC and RSDC are unbiased but are only 2% and 4% efficient, respectively. Note that the efficiency of the degree correction techniques depends on the degree distribution of the graph. They will be more efficient on low-degree graphs and less efficient on high-degree graphs.

IV. Related Work

Sampling from a class of graphs has been well studied in the graph theory literature [5], [11], where they define a class of graphs sharing some property (*e.g.*, degree distribution) and prove that a particular random algorithm can generate all graphs in the class. Cooper *et al.* [7] use this approach to show their algorithm for overlay construction generates graphs with good properties. Our work is quite different; instead of *sampling a graph from a class of graphs* our concern is *sampling peers from a particular graph*.

Others use sampling to extract information from graphs, *e.g.*, sampling a representative subgraph from a large, intractable graph, while maintaining properties of the original [12], [13], [20]. Others use sampling as a component of efficient, randomized algorithms [23]. However, these studies rely on having knowledge of the graph in advance. Our problem is quite different because we have imperfect information.

A closely related problem to ours is sampling Internet routers by running traceroute from a few hosts to many addresses. Using simulation [14] and analysis [2], research shows that traceroute samples can lead to the appearance of a power-law degree distribution regardless of the true distribution. Like our study, they evaluate sampling when there is imperfect information. Our study differs in its basic operation for graph-exploration. In their study, the basic operation is “What is the path to this address?”. In our study, the basic operation is “What are the neighbors of this peer?”.

Another closely related problem is selecting web pages uniformly at random from the set of all web pages [3], [9], [18]. Web pages naturally form a graph, with hyper-links forming edges between pages. Unlike peer-to-peer networks, the graph is *directed* and only outgoing links are easily discovered. Much of the work on sampling web pages therefore focuses on estimating the number of incoming links, to facilitate degree correction. Unlike peers in peer-to-peer systems, web pages are generally regarded as relatively stable, and temporal causes of sampling bias have not been considered in the web context.

Several properties of random walks have been extensively studied analytically [15], such as the access time, cover time, and mixing time. While these properties have many useful applications,to our knowledge the application of random walks as a method of selecting nodes uniformly at random from an unknown graph has not been well studied. Additionally, analytical techniques are only useful for examining classes of graphs which can be expressed mathematically, while in our work we also examine a graph (the Gnutella topology) that was captured empirically.

A number of papers [6], [8], [16] have made use of random walks as a basis for searching unstructured P2P networks. However, searching simply requires locating a certain piece of data *anywhere* along the walk, and is not particularly concerned if some nodes are preferred over others. Gkantsidis *et al.* additionally use random walks as a component of their overlay-construction algorithm.

V. Conclusions and Future Work

In this paper we have explored several techniques for sampling from P2P systems. One of our contributions is to show that unbiased sampling must allow the same peer to be selected multiple times to avoid bias correlated with peer sessions lengths.

We simulated each technique over ordinary random graphs as well as a real Gnutella topology and evaluated how much bias and correlation they introduce as well as their efficiency. We found that the commonly used BFS technique, while efficient, introduces significant sampling bias. Conducting random walks is also significantly biased and additionally is inefficient. The random stroll technique corrects the inefficiency, but remains significantly biased. Each of these techniques are biased due to the influence of the degree distribution. We describe a "degree correction" modification to the random walk and random stroll techniques that corrects the bias, resulting in samples that appear just as accurate as using an oracle. However, there is a significant decrease in efficiency when using these techniques.

In our ongoing work, we are extending our study to include additional types of random graphs, such as power-law random graphs and small-world graphs. By comparing and contrasting the performance of different techniques in different settings, we can gain a better understanding of the most efficient techniques to yield unbiased samples. Additionally, we are exploring techniques for estimating global properties, such as the number of peers in a P2P system or the diameter of an overlay network by exploring only a fraction of the graph.

References

[1] slyck.com. `http://www.slyck.com`, 2005.

[2] D. Achlioptas, A. Clauset, D. Kempe, and C. Moore. On the Bias of Traceroute Sampling; or, Power-law Degree Distributions in Regular Graphs. In *Symposium on Theory of Computing*, 2005.

[3] Z. Bar-Yossef, A. Berg, S. Chien, J. Fakcharoenphol, and D. Weitz. Approximating Aggregate Queries about Web Pages via Random Walks. In *International Conference on Very Large Databases*, 2000.

[4] R. Bhagwan, S. Savage, and G. Voelker. Understanding Availability. In *International Workshop on Peer-to-Peer Systems*, 2003.

[5] B. Bollobás. A probabilistic proof of an asymptotic formula for the number of labelled regular graphs. *European Journal of Combinatorics*, 1, 1980.

[6] Y. Chawathe, S. Ratnasamy, and L. Breslau. Making Gnutella-like P2P Systems Scalable. In *SIGCOMM*, 2003.

[7] C. Cooper, M. Dyer, and C. Greenhill. Sampling regular graphs and a peer-to-peer network. In *Symposium on Discrete Algorithms*, 2005.

[8] C. Gkantsidis, M. Mihail, and A. Saberi. Random Walks in Peer-to-Peer Networks. In *INFOCOM*, 2004.

[9] M. Henzinger, A. Heydon, M. Mitzenmacher, and M. Najork. On Near-Uniform URL Sampling. In *International World Wide Web Conference*, 2001.

[10] M. Izal, G. Urvoy-Keller, E. W. Biersack, P. A. Felber, A. A. Hamra, and L. Garces-Erice. Dissecting BitTorrent: Five Months in a Torrent's Lifetime. In *PAM*, 2004.

[11] M. Jerrum and A. Sinclair. Fast uniform generation of regular graphs. *Theoretical Computer Science*, 73, 1990.

[12] V. Krishnamurthy, M. Faloutsos, M. Chrobak, L. Lao, J.-H. Cui, and A. G. Percus. Reducing Large Internet Topologies for Faster Simulations. In *IFIP Networking*, 2005.

[13] V. Krishnamurthy, J. Sun, M. Faloutsos, and S. Tauro. Sampling Internet Topologies: How Small Can We Go? In *International Conference on Internet Computing*, 2003.

[14] A. Lakhina, J. W. Byers, M. Crovella, and P. Xie. Sampling Biases in IP Topology Measurements. In *INFOCOM*, 2003.

[15] L. Lovász. Random walks on graphs: A survey. *Combinatorics: Paul Erdös is Eighty*, 2, 1993.

[16] Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker. Search and Replication in Unstructured Peer-to-Peer Networks. In *International Conference on Supercomputing*, 2002.

[17] J. Pouwelse, P. Garbacki, D. Epema, and H. Sips. The Bittorrent P2P File-sharing System: Measurements and Analysis. In *International Workshop on Peer-to-Peer Systems (IPTPS)*, 2005.

[18] P. Rusmevichientong, D. M. Pennock, S. Lawrence, and C. L. Giles. Methods for Sampling Pages Uniformly from the World Wide Web. In *AAAI Fall Symposium on Using Uncertainty Within Computation*, 2001.

[19] S. Saroiu, P. K. Gummadi, and S. D. Gribble. Measuring and Analyzing the Characteristics of Napster and Gnutella Hosts. *Multimedia Systems Journal*, 9(2), 2003.

[20] M. P. H. Stumpf, C. Wiuf, and R. M. May. Subnets of scale-free networks are not scale-free: Sampling properties of networks. *Proceedings of the National Academy of Sciences*, 102(12), 2005.

[21] D. Stutzbach and R. Rejaie. Capturing Accurate Snapshots of the Gnutella Network. In *Global Internet Symposium*, 2005.

[22] D. Stutzbach, R. Rejaie, and S. Sen. Characterizing Unstructured Overlay Topologies in Modern P2P File-Sharing Systems. In *Internet Measurement Conference*, 2005.

[23] A. A. Tsay, W. S. Lovejoy, and D. R. Karger. Random Sampling in Cut, Flow, and Network Design Problems. *Mathematics of Operations Research*, 24(2), 1999.

[24] S. Zhao, D. Stutzbach, and R. Rejaie. Characterizing Files in the Modern Gnutella Network: A Measurement Study. In *Multimedia Computing and Networking*, 2006.