On Unbiased Sampling for Unstructured Peer-to-Peer Networks

Daniel Stutzbach, Reza Rejaie, Nick Duffield, Subhabrata Sen, and Walter Willinger
daniel@stutzbachenterprises.com, reza@cs.uoregon.edu, {duffield,sen,walter}@research.att.com

***Abstract*— This paper presents a detailed examination of how the dynamic and heterogeneous nature of real-world peer-to-peer systems can introduce bias into the selection of representative samples of peer properties (*e.g.*, degree, link bandwidth, number of files shared). We propose the *Metropolized Random Walk with Backtracking (MRWB)* as a viable and promising technique for collecting nearly unbiased samples and conduct an extensive simulation study to demonstrate that our technique works well for a wide variety of commonly-encountered peer-to-peer network conditions. We have implemented the MRWB algorithm for selecting peer addresses uniformly at random into a tool called `ion-sampler`. Using the Gnutella network, we empirically show that `ion-sampler` yields more accurate samples than tools that rely on commonly-used sampling techniques and results in dramatic improvements in efficiency and scalability compared to performing a full crawl.**

***Index Terms*— Peer-to-peer, Sampling**

I. Introduction

The popularity and wide-spread use of peer-to-peer systems has motivated numerous empirical studies aimed at providing a better understanding of the properties of deployed peer-to-peer systems. However, due to the large scale and highly dynamic nature of many of these systems, directly measuring the quantities of interest on every peer is prohibitively expensive. Sampling is a natural approach for learning about these systems using light-weight data collection, but commonly-used sampling techniques for measuring peer-to-peer systems tend to introduce considerable bias for two reasons. 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 nature of the overlay topology can lead to bias towards high-degree peers.

In this paper, we are concerned with the basic objective of devising an unbiased sampling method, *i.e.*, one which selects any of the present peers with equal probability. The addresses of the resulting peers may then be used as input to another measurement tool to collect data on particular peer properties (*e.g.*, degree, link bandwidth, number of files shared). The focus of our work is on *unstructured* P2P systems, where peers select neighbors through a predominantly random process. Most popular P2P systems in use today belong to this unstructured category. For structured P2P systems such as Chord [1] and CAN [2], knowledge of the structure significantly facilitates unbiased sampling as we discuss in Section VII.

Achieving the basic objective of selecting any of the peers present with equal probability is non-trivial when the structure of the peer-to-peer system changes during the measurements. First-generation measurement studies of P2P systems typically relied on ad-hoc sampling techniques (*e.g.*, [3], [4]) and provided valuable information concerning basic system behavior. However, lacking any critical assessment of the quality of these sampling techniques, the measurements resulting from these studies may be biased and consequently our understanding of P2P systems may be incorrect or misleading.

The main contributions of this paper are *(i)* a detailed examination of the ways that the topological and temporal qualities of peer-to-peer systems (e.g., churn [5]) can introduce bias, *(ii)* an in-depth exploration of the applicability of a sampling technique called the *Metropolized Random Walk with Backtracking (MRWB)*, representing a variation of the Metropolis–Hastings method [6]–[8], and *(iii)* an implementation of the MRWB algorithm into a tool called `ion-sampler`. While sampling techniques based on the original Metropolis–Hastings method have been considered earlier (e.g., see Awan *et al.* [9] and Bar-Yossef and Gurevich [10]), we show that in the context of unstructured P2P systems, our modification of the basic Metropolis–Hastings method results in nearly unbiased samples under a wide variety of commonly encountered peer-to-peer network conditions.

The proposed MRWB algorithm assumes that the P2P system provides some mechanism to query a peer for a list of its neighbors – a capability provided by most widely deployed P2P systems. Our evaluations of the `ion-sampler` tool shows that the MRWB algorithm yields more accurate samples than previously considered sampling techniques. We quantify the observed differences, explore underlying causes, address the tool's efficiency and scalability, and discuss the implications on accurate inference of P2P properties and high-fidelity modeling of P2P systems. While our focus is on P2P networks, many of our results apply to any large, dynamic, undirected graph where nodes may be queried for a list of their neighbors.

After discussing related work and alternative sampling techniques in Section 2, we build on our earlier formulation in [11] and focus on sampling techniques that select a set of peers uniformly at random from all the peers present in the overlay

An earlier version of this paper appeared in the proceedings of the ACM SIGCOMM Internet Measurement Conference 2006.

Daniel Stutzbach is with Stutzbach Enterprises, LLC.

Reza Rejaie is with the Department of Computer Science at the University of Oregon.

Nick Duffield, Subhabrata Sen and Walter Willinger are with AT&T Labs–Research, Florham Park, New Jersey.

and then gather data about the desired properties from those peers. While it is relatively straightforward to choose peers uniformly at random in a static and known environment, it poses considerable problems in a highly dynamic setting like P2P systems, which can easily lead to significant measurement bias for two reasons.

The first cause of sampling bias derives from the temporal dynamics of these systems, whereby new peers can arrive and existing peers can depart at any time. Locating a set of peers and measuring their properties takes time, and during that time the peer constituency is likely to change. In Section III, 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 relates to the connectivity structure of P2P systems. As a sampling program explores a given topological structure, each traversed link is more likely to lead to a high-degree peer than a low-degree peer, significantly biasing peer selection. We describe and evaluate different techniques for traversing static overlays to select peers in Section IV and find that the Metropolized Random Walk (MRW) collects unbiased samples.

In Section V, we adapt MRW for dynamic overlays by adding backtracking and demonstrate its viability and effectiveness when the causes for both temporal and topological bias are present. We show via simulations that the MRWB technique works well and produces nearly unbiased samples under a variety of circumstances commonly encountered in actual P2P systems.

Finally, in Section 6 we describe the implementation of the `ion-sampler` tool based on the proposed MRWB algorithm and empirically evaluate its accuracy and efficiency through comparison with complete snapshots of Gnutella taken with Cruiser [12], as well as with results obtained from previously used, more ad-hoc, sampling techniques. Section VII discusses some important questions such as how many samples to collect and outlines a practical solution to obtaining unbiased samples for structured P2P systems. Section VIII concludes the paper by summarizing our findings and plans for future work.

II. Related Work

A. Graph sampling

The phrase "graph sampling" means different things in different contexts. For example, sampling from a class of graphs has been well studied in the graph theory literature [13], [14], where the main objective is to prove that for a class of graphs sharing some property (*e.g.*, same node degree distribution), a given random algorithm is capable of generating all graphs in the class. Cooper *et al.* [15] used this approach to show that their algorithm for overlay construction generates graphs with good properties. Our objective is quite different; instead of *sampling a graph from a class of graphs* our concern is *sampling peers (i.e., vertices) from a largely unknown and dynamically changing graph*.

Others have used sampling to extract information about graphs (*e.g.*, selecting representative subgraphs from a large, intractable graph) while maintaining properties of the original structure [16]–[18]. Sampling is also frequently used as a component of efficient, randomized algorithms [19]. However, these studies assume complete knowledge of the graphs in question. Our problem is quite different in that we do not know the graphs in advance.

A closely related problem to ours is sampling Internet routers by running traceroute from a few hosts to many destinations for the purpose of discovering the Internet's router-level topology. Using simulation [20] and analysis [21], research has shown that traceroute measurements can result in measurement bias in the sense that the obtained samples support the inference of power law-type degree distributions irrespective of the true nature of the underlying degree distribution. A common feature of our work and the study of the traceroute technique [20], [21] is that both efforts require an evaluation of sampling techniques without complete knowledge of the true nature of the underlying connectivity structure. However, exploring the router topology and P2P topologies differ in their basic operations for graph-exploration. In the case of traceroute, the basic operation is "What is the path to this destination?" In P2P networks, the basic operation is "What are the neighbors of this peer?" In addition, the Internet's router-level topology changes at a much slower rate than the overlay topology of P2P networks.

Another closely related problem is selecting Web pages uniformly at random from the set of all Web pages [22], [23]. Web pages naturally form a graph, with hyper-links forming edges between pages. Unlike unstructured peer-to-peer networks, the Web 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, not much is known about the temporal stability of Web pages, and temporal causes of sampling bias have received little attention in past measurement studies of the Web.

B. Random walk-based sampling of graphs

A popular technique for exploring connectivity structures consists of performing random walks on graphs. Several properties of random walks on graphs have been extensively studied analytically [24], such as the access time, cover time, and mixing time. While these properties have many useful applications, they are, in general, only well-defined for static graphs. To our knowledge the application of random walks as a method of selecting nodes uniformly at random from a *dynamically changing* graph has not been studied.

A number of papers [25]–[28] 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. Some studies [27], [28] additionally use random walks as a component of their overlay-construction algorithm.

Two papers that are closely related to our random walk-based sampling approach are by Awan *et al.* [9] and Bar-Yossef and Gurevich [10]. While the former also address the problem of gathering uniform samples from peer-to-peer networks, the

latter are concerned with uniform sampling from a search engine's index. Both works examine several random walk techniques, including the Metropolis–Hastings method, but assume an underlying graph structure that is not dynamically changing. In addition to evaluating their techniques empirically for static power-law graphs, the approach proposed by Awan *et al.* [9] also requires special underlying support from the peer-to-peer application. In contrast, we implement the Metropolis–Hastings method in such a way that it relies only on the ability to discover a peer's neighbors, a simple primitive operation commonly found in existing peer-to-peer networks. Moreover, we introduce backtracking to cope with departed peers and conduct a much more extensive evaluation of the proposed MRWB method. Specifically, we generalize our formulation reported in [11] by evaluating MRWB over dynamically changing graphs with a variety of topological properties. We also perform empirical validations over an actual P2P network.

C. Sampling in hidden populations

The problem of obtaining accurate estimates of the number of peers in an unstructured P2P network that have a certain property can also be viewed as a problem in studying the sizes of *hidden populations*. Following Salganik [29], a population is called "hidden" if there is no central directory of all population members, such that samples may only be gathered through referrals from existing samples. This situation often arises when public acknowledgment of membership has repercussions (*e.g.*, injection drug users [30]), but also arises if the target population is difficult to distinguish from the population as a whole (*e.g.*, jazz musicians [29]). Peers in P2P networks are hidden because there is no central repository we can query for a list of all peers. Peers must be discovered by querying other peers for a list of neighbors.

Proposed methods in the social and statistical sciences for studying hidden populations include *snowball sampling* [31], *key informant sampling* [32], and *targeted sampling* [33]. While these methods gather an adequate number of samples, they are notoriously biased. More recently, Heckathorn [30] (see also [29], [34]) proposed *respondent-driven sampling*, a snowball-type method for sampling and estimation in hidden populations. Respondent-driven sampling first uses the sample to make inferences about the underlying network structure. In a second step, these network-related estimates are used to derive the proportions of the various subpopulations of interest. Salganik *et al.* [29], [34] show that under quite general assumptions, respondent-driven sampling yields estimates for the sizes of subpopulations that are asymptotically unbiased, no matter how the seeds were chosen.

Unfortunately, respondent-driven sampling has only been studied in the context where the social network is static and does not change with time. To the best of our knowledge, the accuracy of respondent-driven sampling in situations where the underlying network structure is changing dynamically (e.g., unstructured P2P systems) has not been considered in the existing sampling literature.

D. Dynamic graphs

While graph theory has been largely concerned with studying and discovering properties of static connectivity structures, many real-world networks evolve over time, for example via node/edge addition and/or deletion. In fact, many large-scale networks that arise in the context of the Internet (e.g., WWW, P2P systems) are extremely dynamic and create havoc for graph algorithms that have been designed with static or only very slowly changing network structures in mind. Furthermore, the development of mathematical models for evolving graphs is still at an early stage and is largely concerned with generative models that are capable of reproducing certain observed properties of evolving graphs. For example, recent work by Leskovec *et al.* [35] focuses on empirically observed properties such as densification (*i.e.*, networks become denser over time) and shrinking diameter (*i.e.*, as networks grow, their diameter decreases) and on new graph generators that account for these properties. However, the graphs they examine are not P2P networks and their properties are by and large inconsistent with the design and usage of measured P2P networks (e.g., see [5]). Hence, the dynamic graph models proposed in [35] are not appropriate for our purpose, and neither are the evolving graph models specifically designed to describe the Web graph (e.g., see [36] and references therein).

III. Sampling 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 set of time-indexed graphs, $G_t = G(V_t, E_t)$. The most common approach for sampling from this set of graphs is to define a measurement window, $[t_0, t_0 + \Delta]$, and select peers uniformly at random from the set of peers who are present at any time during the window: $V_{t_0,t_0+\Delta} = \bigcup_{t=t_0}^{t_0+\Delta} V_t$. Thus, it does not distinguish between occurrences of the same peer at different times.

This approach is appropriate if peer session lengths are exponentially distributed (*i.e.*, memoryless). However, existing measurement studies [3], [5], [37], [38] 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 the measurement window, 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. Thus, we distinguish between occurrences of the same peer at different times: samples $v_{i,t}$ and $v_{i,t'}$ gathered at distinct times $t \neq t'$ are viewed as distinct, even when 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.* Using the formulation $v_{i,t} \in V_t,\ t \in [t_0, t_0 + \Delta]$, the 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 make this distinction.

Returning to our simple example, our 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.

If the measurement window (Δ) is sufficiently small, such that the distribution of the property under consideration does not change significantly during the measurement window, then we may relax the constraint of choosing t uniformly at random from $[t_0, t_0 + \Delta]$.

We still have the significant problem of selecting a peer uniformly at random from those present at a particular time. We begin to address this problem in the next section.

IV. Sampling from Static Graphs

We now turn our attention to topological causes of bias. Towards this end, we momentarily set aside the temporal issues by assuming a static, unchanging graph. The selection process begins with knowledge of one peer (vertex) and progressively queries peers for a list of neighbors. The goal is to select peers uniformly at random. In any graph-exploration problem, we have a set of visited peers (vertices) and a front of unexplored neighboring peers. There are two ways in which algorithms differ: *(i)* how to chose the next peer to explore, and *(ii)* which subset of the explored peers to select as samples. Prior studies use simple breadth-first or depth-first approaches to explore the graph and select all explored peers. These approaches suffer from several problems:

- The discovered peers are correlated by their neighbor relationship.
- Peers with higher degree are more likely to be selected.
- Because they never visit the same peer twice, they will introduce bias when used in a dynamic setting as described in Section III.

Random Walks: A better candidate solution is the random walk, which has been extensively studied in the graph theory literature (for an excellent survey see [24]). We briefly summarize the key terminology and results relevant to sampling. The transition matrix $P(x, y)$ describes the probability of transitioning to peer y if the walk is currently at peer x:

$$P(x,y) = \begin{cases} \frac{1}{\text{degree}(x)} & y \text{ is a neighbor of x,} \\ 0 & \text{otherwise} \end{cases}$$

If the vector v describes the probability of currently being at each peer, then the vector $v' = vP$ describes the probability after taking one additional step. Likewise, vP^r describes the probability after taking r steps. As long as the graph is connected and not bipartite, the probability of being at any particular node, x, converges to a *stationary distribution*:

$$\pi(x) = \lim_{r\to\infty} (vP^r)(x) = \frac{\text{degree}(x)}{2 \cdot |E|}$$

In other words, if we select a peer as a sample every r steps, for sufficiently large r, we have the following good properties:

- The information stored in the starting vector, v, is lost, through the repeated selection of random neighbors. Therefore, there is no correlation between selected peers. Alternately, we may start many walks in parallel. In either cases, after r steps, the selection is independent of the origin.
- While the stationary distribution, $\pi(x)$, is biased towards peers with high degree, the bias is precisely known, allowing us to correct it.
- Random walks may visit the same peer twice, which lends itself better to a dynamic setting as described in Section III.

In practice, r need not be exceptionally large. For graphs where the edges have a strong random component (*e.g.*, small-world graphs such as peer-to-peer networks), it is sufficient that the number of steps exceed the log of the population size, *i.e.*, $r \geq O(\log |V|)$.

Adjusting for degree bias: To correct for the bias towards high degree peers, we make use of the Metropolis–Hastings method for Markov Chains. Random walks on a graph are a special case of Markov Chains. In a regular random walk, the transition matrix $P(x, y)$ leads to the stationary distribution $\pi(x)$, as described above. We would like to choose a new transition matrix, $Q(x, y)$, to produce a different stationary distribution, $\mu(x)$. Specifically, we desire $\mu(x)$ to be the uniform distribution so that all peers are equally likely to be at the end of the walk. Metropolis–Hastings [6]–[8] provides us with the desired $Q(x, y)$:

$$Q(x,y) = \begin{cases} P(x,y) \min\left(\frac{\mu(y)P(y,x)}{\mu(x)P(x,y)}, 1\right) & \text{if } x \neq y, \\ 1 - \sum_{z\neq x} Q(x,z) & \text{if } x = y \end{cases}$$

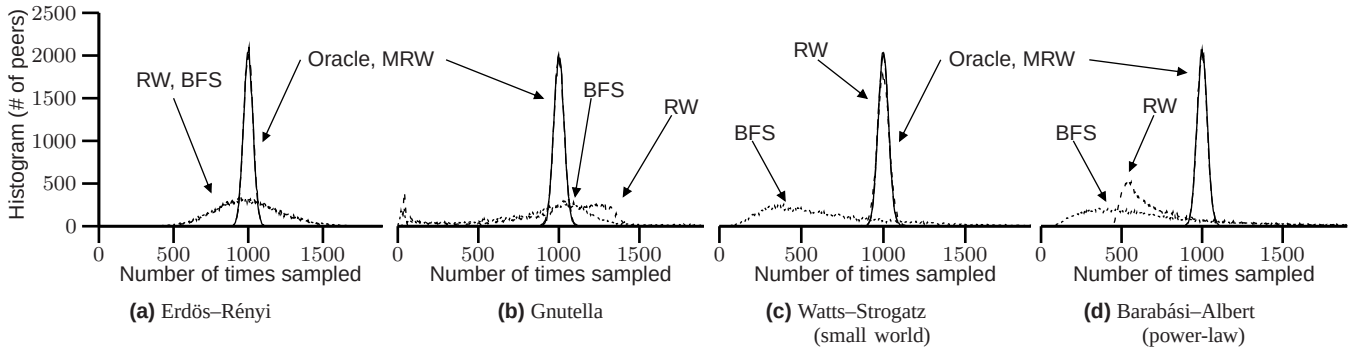
Equivalently, to take a step from peer x, select a neighbor y of x as normal (*i.e.*, with probability $P(x, y)$). Then, with probability $\min\left(\frac{\mu(y)P(y,x)}{\mu(x)P(x,y)}, 1\right)$, accept the move. Otherwise, return to x (*i.e.*, with probability $1 - \sum_{z\neq x} Q(x, z)$).

To collect uniform samples, we have $\frac{\mu(y)}{\mu(x)} = 1$, so the move-acceptance probability becomes:

$$\min\left(\frac{\mu(y)P(y,x)}{\mu(x)P(x,y)}, 1\right) = \min\left(\frac{\text{degree}(x)}{\text{degree}(y)}, 1\right)$$

Therefore, our algorithm for selecting the next step from some peer x is as follows:

- Select a neighbor y of x uniformly at random.
- Query y for a list of its neighbors, to determine its degree.
- Generate a random value, p, uniformly between 0 and 1.



(a) Erdös–Rényi **(b)** Gnutella **(c)** Watts–Strogatz (small world) **(d)** Barabási–Albert (power-law)

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.

	Erdös–Rényi	Gnutella	Watts–Strogatz	Barabási–Albert
Breadth-First Search	$4.54 \cdot 10^{-4}$	$2.73 \cdot 10^{-3}$	4.73^{-3}	$2.77 \cdot 10^{-3}$
Random Walk	$3.18 \cdot 10^{-4}$	$1.57 \cdot 10^{-3}$	7.64^{-5}	$2.84 \cdot 10^{-3}$
Metropolis–Hastings	$5.97 \cdot 10^{-5}$	$5.79 \cdot 10^{-5}$	6.08^{-5}	$5.22 \cdot 10^{-5}$

TABLE I: Kolmogorov-Smirnov test statistic for techniques over static graphs. Values above $1.07 \cdot 10^{-4}$ lie in the rejection region at the 5% level.

- If $p \leq \frac{\text{degree}(x)}{\text{degree}(y)}$, y is the next step.
- Otherwise, remain at x as the next step.

We call this the Metropolized Random Walk (MRW). Qualitatively, the effect is to suppress the rate of transition to peers of higher degree, resulting in selecting each peer with equal probability.

Evaluation: Although [6] provides a proof of correctness for the Metropolis–Hastings method, to ensure the correctness of our implementation we conduct evaluations through simulation over static graphs. This additionally provides the opportunity to compare MRW with conventional techniques such as Breadth-First Search (BFS) or naive random walks (RW) with no adjustments for degree bias.

To evaluate a technique, we use it to collect a large number of sample vertices from a graph, then perform a goodness-of-fit test against the uniform distribution. For Breadth-First Search, we simulate typical usage by running it to gather a batch of 1,000 peers. When one batch of samples is collected, the process is reset and begins anew at a different starting point. To ensure robustness with respect to different kinds of connectivity structures, we examine each technique over several types of graphs as follows:

- **Erdös–Rényi:** The simplest variety of random graphs
- **Watts–Strogatz:** "Small world" graphs with high clustering and low path lengths
- **Barabási–Albert:** Graphs with extreme degree distributions, also known as power-law or scale-free graphs
- **Gnutella:** Snapshots of the Gnutella ultrapeer topology, captured in our earlier work [39]

To make the results more comparable, the number of vertices ($|V| = 161,680$) and edges ($|E| = 1,946,596$) in each graph are approximately the same.[1] Table I presents the results of the goodness-of-fit tests after collecting $1000 \cdot |V|$ samples, showing that Metropolis–Hastings appears to generate uniform samples over each type of graph, while the other techniques fail to do so by a wide margin.

Figure 1 explores the results visually, by plotting the number of times each peer is selected. If we select $k \cdot |V|$ samples, the typical node should be selected k times, with other nodes being selected close to k times approximately following a normal distribution with variance k.[2] We used $k = 1,000$ samples. We also include an "Oracle" technique, which selects peers uniformly at random using global information. The Metropolis–Hastings results are virtually identical to the Oracle, while the other techniques select many peers much more and much less than k times. In the Gnutella, Watts–Strogatz, and Barabási–Albert graphs, Breadth-First Search exhibits a few vertices that are selected a large number of times ($> 10,000$). The (not-adjusted) Random Walk (RW) method has similarly selected a few vertices an exceptionally large number of times in the Gnutella and Barabási–Albert models. The Oracle and MRW, by contrast, did not select any vertex more than around 1,300 times.

In summary, the Metropolis–Hastings method selects peers uniformly at random from a static graph. The next section examines the additional complexities when selecting from a dynamic graph, introduces appropriate modifications, and evaluates the algorithm's performance.

V. Sampling from Dynamic Graphs

Section III set aside topological issues and examined the dynamic aspects of sampling. Section IV set aside temporal issues and examined the topological aspects of sampling. This section examines the unique problems that arise when both temporal and topological difficulties are present.

Our hypothesis is that a Metropolis–Hastings random walk will yield approximately unbiased samples even in a dynamic

[1] Erdös–Rényi graphs are generated based on some probability p that any edge may exist. We set $p = \frac{2|E|}{|V| \cdot (|V|-1)}$ so that there will be close to $|E|$ edges, though the exact value may vary slightly. The Watts–Strogatz model require that $|E|$ be evenly divisible by $|V|$, so in that model we use $|E| = 1,940,160$.

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

environment. Simulation results testing this hypothesis are later in this section and empirical tests are in the next section. The fundamental assumption of Metropolis–Hastings is that the frequency of visiting a peer is proportional to the peer's degree. This assumption will be approximately correct if peer relationships change only slightly during the walk. On one extreme, if the entire walk completes before any graph changes occur, then the problem reduces to the static case. If a single edge is removed mid-walk, the probability of selecting the two affected peers is not significantly affected, unless those peers have very few edges. If many edges are added and removed during a random walk, but the degree of each peer does not change significantly, we would also expect that the probability of selecting each peer will not change significantly. In peer-to-peer systems, each peer actively tries to maintain a number of connections within a certain range, so we have reason to believe that the degree of each peer will be relatively stable in practice. On the other hand, it is quite possible that in a highly dynamic environment, or for certain degree distributions, the assumptions of Metropolis–Hastings are grossly violated and it fails to gather approximately unbiased samples.

The fundamental question we attempt to answer in this section is: *Under what conditions does the Metropolis–Hastings random walk fail to gather approximately unbiased samples?* If there is any bias in the samples, the bias will be tied to some property that interacts with the walk. Put another way, if there were no properties that interacted with the walk, then the walking process behaves as it would on a static graph, for which we have a proof from graph theory. Therefore, we are only worried about properties which cause the walk to behave differently. We identify the following three fundamental properties that interact with the walk:

- **Degree:** the number of neighbors of each peer. The Metropolis–Hastings method is a modification of a regular random walk in order to correct for degree-bias as described in Section IV. It assumes a fixed relationship between degree and the probability of visiting a peer. If the Metropolis–Hastings assumptions are invalid, the degree-correction may not operate correctly, introducing a bias correlated with degree.
- **Session lengths:** how long peers remain in the system. Section III showed how sampling may result in a bias based on session length. If the walk is more likely to select either short-lived or long-lived peers, there will be a bias correlated with session length.
- **Query latency:** how long it takes the sampler to query a peer for a list of its neighbors. In a static environment the only notion of time is the number of steps taken by the walk. In a dynamic environment, each step requires querying a peer, and some peers will respond more quickly than others. This could lead to a bias correlated with the query latency. In our simulations, we model the query latency as twice the round-trip time between the sampling node and the peer being queried.[3]

For other peer properties, sampling bias can only arise if the desired property is correlated with a fundamental properties and that fundamental property exhibits bias. For example, when sampling the number of files shared by each peer, there may be sampling bias if the number of files is correlated with session length *and* sampling is biased with respect to session length. One could also imagine the number of files being correlated with query latency (which is very loosely related to the peer bandwidth). However, sampling the number of shared files cannot be biased independently, as it does not interact with the walk. To show that sampling is unbiased for any property, it is sufficient to show that it is unbiased for the fundamental properties that interact with the sampling technique.

[3] $\frac{1}{2}$ RTT for the SYN, $\frac{1}{2}$ RTT for the SYN-ACK, $\frac{1}{2}$ RTT for the ACK and the request, and $\frac{1}{2}$ RTT for the reply.

A. Coping with Departing Peers

Departing peers introduce an additional practical consideration. The walk may try to query a peer that is no longer present–a case where the behavior of the ordinary random walk algorithm is undefined. We employ a simple adaptation to mimic an ordinary random walk on a static graph as closely as possible, by maintaining a stack of visited peers. When the walk chooses a new peer to query, we push the peer's address on the stack. If the query times out, we pop the address off the stack, and choose a new neighbor of the peer that is now on top of the stack. If all of a peer's neighbors time out, we re-query that peer to get a fresh list of its neighbors. If the re-query also times out, we pop that peer from the stack as well, and so on. If the stack underflows, we consider the walk a failure. We do not count timed-out peers as a hop for the purposes of measuring the length of the walk. We call this adaptation of the MRW sampling technique the *Metropolized Random Walk with Backtracking (MRWB)* method for sampling from dynamic graphs. Note that when applied in a static environment, this method reduces to MRW.

B. Evaluation methodology

In the static case, we can rely on graph theory to prove the accuracy of the MRW technique. Unfortunately, graph theory is not well-suited to the problem of dynamically changing graphs. Therefore, we rely on simulation rather than analysis. We have developed a session-level dynamic overlay simulator that models peer arrivals, departures, latencies, and neighbor connections. We now describe our simulation environment.

The latencies between peers are modeled using values from the King data set [40]. Peers learn about one another using one of several *peer discovery* mechanisms described below. Peers have a target minimum number of connections (*i.e.*, degree) that they attempt to maintain at all times. Whenever they have fewer connections, they open additional connections. We assume connections are TCP and require a 3-way handshake before the connection is fully established, and that peers will time out an attempted connection to a departed peer after 10 seconds. A new peer generates its session length from one of several different session length distributions described below and departs when the session length expires. New peers arrive according to a Poisson process, where we select the mean peer arrival rate based on the session length distribution to achieve a target population size of 100,000 peers.

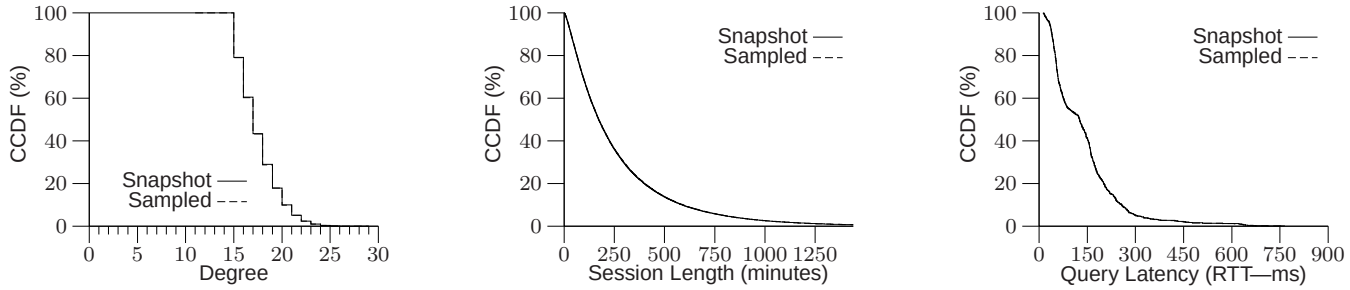


Fig. 2: Comparison of sampled and expected distributions. They are visually indistinguishable.

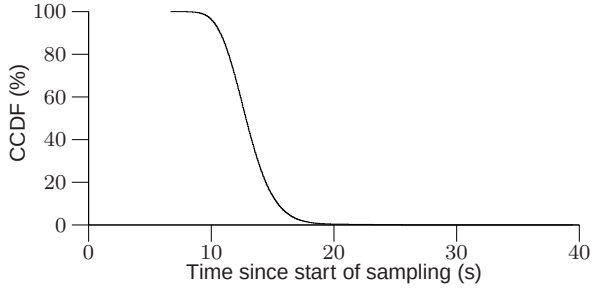


Fig. 3: Distribution of time needed to complete a random walk (simulated)

To query a peer for a list of neighbors, the sampling node must set up a TCP connection, submit its query, and receive a response. The query times out if no response is received after 10 seconds.[4] We run the simulator for a warm-up period to reach steady-state conditions before performing any random walks.

Our goal is to discover if random walks started under identical conditions will select a peer uniformly at random. To evaluate this, we start 100,000 concurrent random walks from a single location. Although started at the same time, the walks will not all complete at the same time.[5] We chose to use 100,000 walks as we believe this is a much larger number of samples than most researchers will use in practice. If there is no discernible bias with 100,000 samples, we can conclude that the tool is unbiased for the purposes of gathering fewer samples (*i.e.*, we cannot get more accuracy by using less precision). Figure 3 shows the distribution of how long walks take to complete in one simulation using 50 hops per walk, illustrating that most walks take 10–20 seconds to complete. In the simulator the walks do not interact or interfere with one another in any way. Each walk ends and collects an independent sample.

As an expected distribution, we capture a perfect snapshot (*i.e.*, using an oracle) at the median walk-completion time, *i.e.*, when 50% of the walks have completed.

C. Evaluation of a base case

Because the potential number of simulation parameters is unbounded, we need a systematic method to intelligently explore the most interesting portion of this parameter space. Towards this end, we begin with a base case of parameters as a starting point and examine the behavior of MRWB under those conditions. In the following subsections, we vary the parameters and explore how the amount of bias varies as a function of each of the parameters. As a base case, we use the following configuration:

Session length distribution:	Weibull($k = 0.59$, $\lambda = 40$)
Target degree:	15
Maximum degree:	30
Peer discovery mechanism:	FIFO

TABLE II: Base Case Configuration

Figure 2 presents the sampled and expected distributions for the three fundamental properties: degree, session length, and query latency. The fact that the sampled and expected distributions are visually indistinguishable demonstrates that the samples are not significantly biased in the base case.

To efficiently examine other cases, we introduce a *summary statistic* to quickly capture the difference between the sampled and expected distributions, and to provide more rigor than a purely visual inspection. For this purpose, we use the Kolmogorov-Smirnov (KS) statistic, D, formally defined as follows. Where $S(x)$ is the sampled cumulative distribution function and $E(x)$ is the expected cumulative distribution function from the perfect snapshot, the KS statistic is:

$$D = \max\left(|S(x) - E(x)|\right)$$

In other words, if we plot the sampled and expected CDFs, D is the maximum vertical distance between them and has a possible range of $[0, 1]$. For Figures 2a, 2b, and 2c, the values of D were 0.0019, 0.0023, and 0.0037, respectively. For comparison, at the $p = 0.05$ significance level, D is 0.0061, for the two-sample KS statistic with 100,000 data points each. However, in practice we do not expect most researchers to gather hundreds of thousands of samples. After all, the initial motivation for sampling is to gather reasonably accurate data at relatively low cost. As a rough rule of thumb, a value of $D \geq 0.1$ is quite bad, corresponding to at least a 10 percentage point difference on a CDF. A value of $D \leq 0.01$ is excellent for most purposes when studying a peer property, corresponding to no more than a 1 percentage point difference on a CDF.

D. Exploring different dynamics

In this section, we examine how the amount of bias changes as we vary the type and rate of dynamics in the system.

[4]The value of 10 seconds was selected based on our experiments in developing a crawler for the Gnutella network in [12].

[5]Each walk ends after the same number of *hops*, but not every hop takes the same amount of *time* due to differences in latencies and due to the occasional timeout.

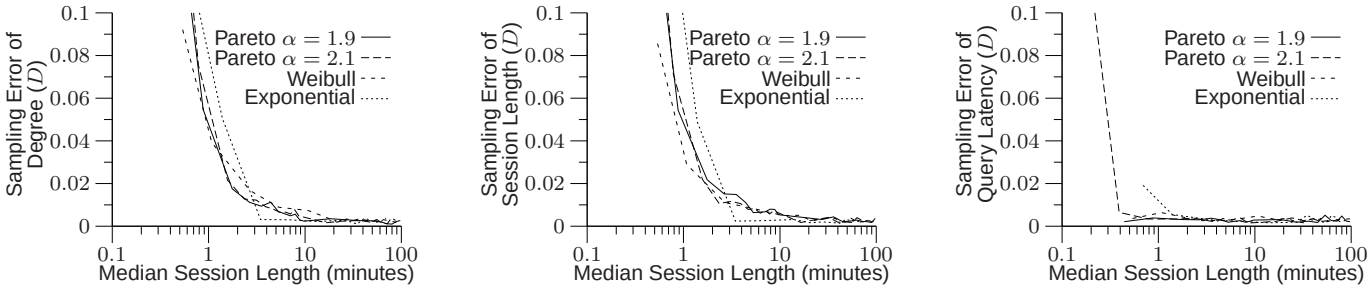


Fig. 4: Sampling error of the three fundamental properties as a function of session-length distribution. Exceptionally heavy churn (median $<$ 1min) introduces error into the sampling process.

We examine different settings of the simulation parameters that affect dynamics, while continuing to use the topological characteristics from our base case (Table II). We would expect that as the rate of peer dynamics increases, the sampling error also increases. The key question is: *How fast can the churn rate be before it causes significant error, and is that likely to occur in practice?*

In this subsection, we present the results of simulations with a wide variety of rates using three different models for session length, as follows:

- **Exponential:** The exponential distribution is a one-parameter distribution (rate λ) that features sessions relatively close together in length. It has been used in many prior simulation and analysis studies of peer-to-peer systems [41]–[43].
- **Pareto:** The Pareto (or power-law) distribution is a two-parameter distribution (shape α, location x_m) that features many short sessions coupled with a few very long sessions. Some prior measurement studies of peer-to-peer systems have suggested that session lengths follow a Pareto distribution [44]–[46]. One difficulty with this model is that x_m is a lower-bound on the session length, and fits of x_m to empirical data are often unreasonably high (*i.e.*, placing a lower bound significantly higher than the median session length reported by other measurement studies). In their insightful analytical study of churn in peer-to-peer systems, Leonard, Rai, and Loguinov [47] instead suggest using a shifted Pareto distribution (shape α, scale β) with $\alpha \approx 2$. We use this shifted Pareto distribution, holding α fixed and varying the scale parameter β. We examine two different α values: $\alpha = 1.9$ (infinite variance) and $\alpha = 2.1$ (finite variance).
- **Weibull:** Our own empirical observations [5] suggest the Weibull distribution (shape k, scale λ) provides a good model of peer session lengths, representing a compromise between the exponential and Pareto distributions. We fix $k = 0.59$ (based on our empirical data) and vary the scale parameter λ.

Figure 4 presents the amount of sampling error (D) as a function of median session length, for the three fundamental properties, with a logarithmic x-axis scale. The figure shows that error is low ($D < 0.01$) over a wide range of session lengths but begins to become significant when the median session length drops below 2 minutes, and exceeds $D = 0.1$ when the median drops below 30 seconds. The type of distribution varies the threshold slightly, but overall does not appear to have a significant impact. To investigate whether the critical threshold is a function of the length of the walk, we ran some simulations using walks of 10,000 hops (which take around one simulated hour to complete). Despite the long duration of these walks, they remained unbiased with $D < 0.003$ for each of the three fundamental properties. *This suggests that the accuracy of MRWB is not adversely affected by a long walk.*

While the median session length reported by measurement studies varies considerably (see [42] for a summary), none report a median below 1 minute and two studies report a median session length of one hour [3], [4]. In summary, these results demonstrate that MRWB can gracefully tolerate peer dynamics. In particular, it performs well over the rate of churn reported in real systems.

E. Exploring different topologies

In this section, we examine different settings of the simulation parameters that directly affect topological structure, while using the dynamic characteristics from our base case (Table II). The Metropolis–Hastings method makes use of the ratio between the degrees of neighboring peers. If this ratio fluctuates dramatically while the walk is conducted, it may introduce significant bias. If peers often have only a few connections, any change in their degree will result in a large percentage-wise change. One key question is therefore: *Does a low target degree lead to sampling bias, and, if so, when is significant bias introduced?*

The degree of peers is controlled by three factors. First, each peer has a *peer discovery mechanism* that enables it to learn the addresses of potential neighbors. The peer discovery mechanism will influence the structure of the topology and, if performing poorly, will limit the ability of peers to establish connections. Second, peers have a *target degree* which they actively try to maintain. If they have fewer neighbors than the target, they open additional connections until they have reached the target. If necessary, they make use of the peer discovery mechanism to locate additional potential neighbors. Finally, peers have a *maximum degree*, which limits the number of neighbors they are willing to accept. If they are at the maximum and another peer contacts them, they refuse the connection. Each of these three factors influences the graph structure, and therefore may affect the walk.

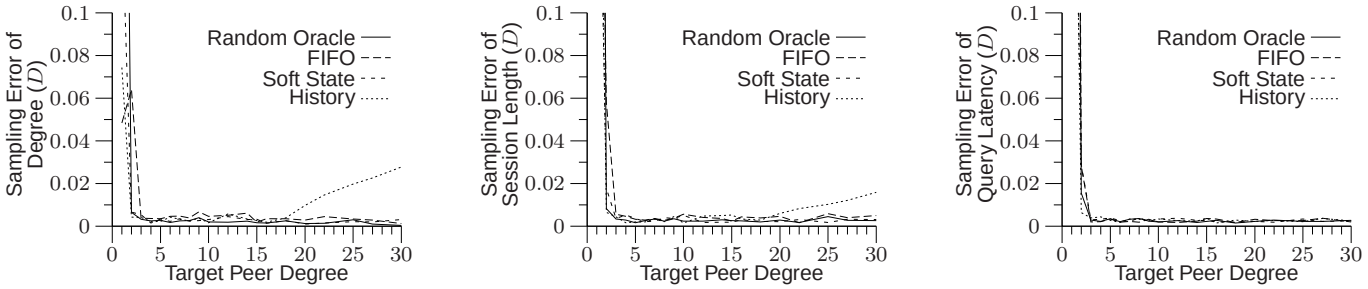


Fig. 5: Sampling error of the three fundamental properties as a function of the number of connections each peer actively attempts to maintain. Low target degree (≤ 2) introduces significant sampling error.

We model four different types of peer discovery mechanisms, based on those found in real systems:

- **Random Oracle:** This is the simplest and most idealistic approach. Peers learn about one another by contacting a rendezvous point that has perfect global knowledge of the system and returns a random set of peers for them to connect to.
- **FIFO:** In this scheme, inspired by the GWebCaches of Gnutella [48], peers contact a rendezvous point which returns a list of the last n peers that contacted the rendezvous, where n is the maximum peer degree.
- **Soft State:** Inspired by the approach of BitTorrent's "trackers", peers contact a rendezvous point that has *imperfect* global knowledge of the system. In addition to contacting the rendezvous point to learn about more peers, every peer periodically (every half hour) contacts the rendezvous point to refresh its state. If a peer fails to make contact for 45 minutes, the rendezvous point removes it from the list of known peers.
- **History:** Many P2P applications connect to the network using addresses they learned during a previous session [49]. A large fraction of these addresses will timeout, but typically enough of the peers will still be active to avoid the need to contact a centralized rendezvous point. As tracking the re-appearance of peers greatly complicates our simulator (as well as greatly increasing the memory requirements), we use a coarse model of the History mechanism. We assume that 90% of connections automatically timeout. The 10% that are given valid addresses are skewed towards peers that have been present for a long time (more than one hour) and represent regular users who might have been present during the peer's last session. While this might be overly pessimistic, it reveals the behavior of MRWB under harsh conditions.

Figure 5 presents the amount of sampling error (D) for the three fundamental properties as a function of the target degree, for each of the peer discovery methods, holding the maximum peer degree fixed at 30 neighbors. It shows that sampling is not significantly biased in any of the three fundamental properties as long as peers attempt to maintain at least three connections. Widely deployed peer-to-peer systems typically maintain dozens of neighbors. Moreover, maintaining fewer than three neighbors per peer almost certainly leads to network fragmentation, and is therefore not a reasonable operating point for peer-to-peer systems.

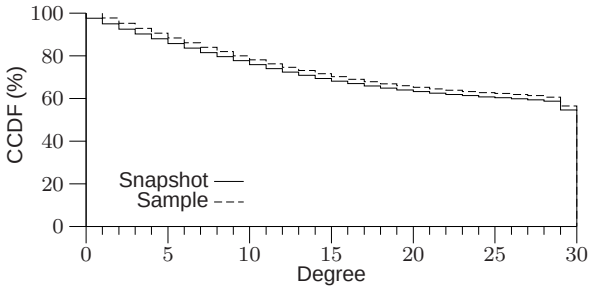


Fig. 6: Comparison of degree distributions using the History mechanism with a target degree of 30. Sampling cannot capture the unconnected peers (degree $= 0$), causing the sampling error observed in Figure 5.

The results for the different peer-discovery mechanisms were similar to one another, except for a small amount of bias observed when using the History mechanism as the target degree approaches the maximum degree (30). To investigate this issue, Figure 6 presents the sampled and expected degree distribution when using the History mechanism with a target degree of 30. The difference between the sampled and expected distributions is due to the 2.4% of peers with a degree of zero. These isolated peers arise in this scenario because the History mechanism has a high failure rate (returning addresses primarily of departed peers), and when a valid address is found, it frequently points to a peer that is already at its connection limit. The zero-degree peers are visible in the snapshot (which uses an oracle to obtain global information), but not to the sampler (since peers with a degree of zero have no neighbors and can never be reached). We do not regard omitting disconnected peers as a serious limitation.

Having explored the effects of lowering the degree, we now explore the effects of increasing it. In Figure 7, we examine sampling error as a function of the maximum degree, with the target degree always set to 15 less than the maximum. There is little error for any setting of the maximum degree.

In summary, the proposed MRWB technique for sampling from dynamic graphs appears unbiased for a range of different topologies (with reasonable degree distributions; *e.g.*, degree ≥ 3), operates correctly for a number of different mechanisms for peer discovery, and is largely insensitive to a wide range of peer dynamics, with the churn rates reported for real systems safely within this range.

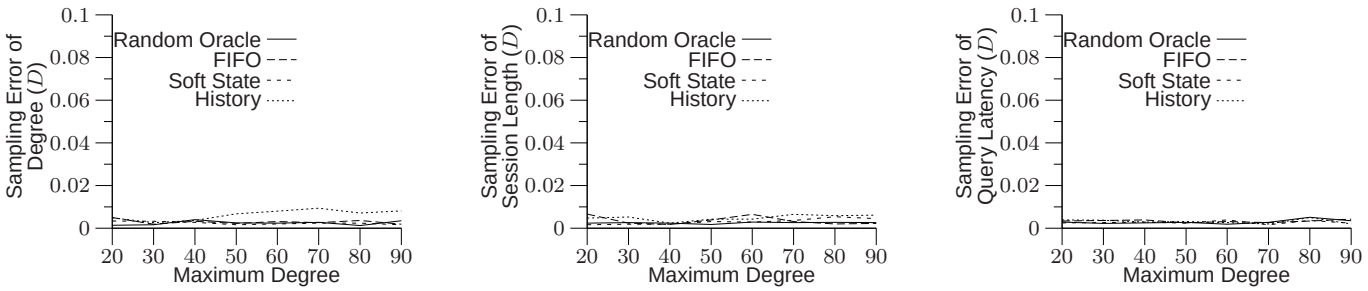


Fig. 7: Sampling error of the three fundamental properties as a function of the maximum number of connections each peer will accept. Each peer actively attempts to maintain $x - 15$ connections.

VI. Empirical Results

In addition to the simulator version, we have implemented the MRWB algorithm for sampling from real peer-to-peer networks into a tool called `ion-sampler`. The following subsections briefly describe the implementation and usage of `ion-sampler` and present empirical experiments to validate its accuracy.

A. Ion-Sampler

The `ion-sampler` tool uses a modular design that accepts plug-ins for new peer-to-peer systems.[6] A plug-in can be written for any peer-to-peer system that allows querying a peer for a list of its neighbors. The `ion-sampler` tool hands IP-address:port pairs to the plug-in, which later returns a list of neighbors or signals that a timeout occurred. The `ion-sampler` tool is responsible for managing the walks. It outputs the samples to standard output, where they may be easily read by another tool that collects the actual measurements. For example, `ion-sampler` could be used with existing measurement tools for measuring bandwidth to estimate the distribution of access link bandwidth in a peer-to-peer system. Listing 1 shows an example of using `ion-sampler` to sample peers from Gnutella.

B. Empirical Validation

Empirical validation is challenging due to the absence of high-quality reference data to compare against. In our earlier work [12], [39], we developed a peer-to-peer crawler called Cruiser that captures the complete overlay topology through exhaustive exploration. We can use these topology snapshots as a point of reference for the degree distribution. Unfortunately, we do not have reliably accurate empirical reference data for session lengths or query latency.

By capturing every peer, Cruiser is immune to sampling difficulties. However, because the network changes as Cruiser operates, its snapshots are slightly distorted [12]. In particular, peers arriving near the start of the crawl are likely to have found additional neighbors by the time Cruiser contacts them. Therefore, we intuitively expect a slight upward bias in Cruiser's observed degree distribution. For this reason, we would not expect a perfect match between Cruiser and sampling, but if the sampling is unbiased we still expect them to be very close. We can view the CCDF version of the degree distribution captured by Cruiser as a close upper-bound on the true degree distribution.

Figure 8 presents a comparison of the degree distribution of reachable ultrapeers in Gnutella, as seen by Cruiser and by the sampling tool (capturing approximately 1,000 samples with $r = 25$ hops). It also includes the results of a short crawl,[7] a sampling technique commonly used in earlier studies (*e.g.*, [3]). We interleaved running these measurement tools to minimize the change in the system between measurements of different tools, in order to make their results comparable.

Examining Figure 8, we see that the full crawl and sampling distributions are quite similar. The sampling tool finds slightly more peers with lower degree, compared to the full crawl, in accordance with our expectations described above. We examined several such pairs of crawling and sampling data and found the same pattern in each pair. By comparison, the short crawl exhibits a substantial bias towards high degree peers relative to both the full crawl and sampling. We computed the KS statistic (D) between each pair of datasets, presented in Table III. Since the full crawl is a close *upper-bound* of the true degree distribution, and since sampling's distribution is *lower*, the error in the sampling distribution relative to the

[6] In fact, it uses the same plug-in architecture as our earlier, heavy-weight tool, Cruiser, which exhaustively crawls peer-to-peer systems to capture topology snapshots.

[7] A "short crawl" is a general term for a progressive exploration of a portion of the graph, such as by using a breadth-first or depth-first search. In this case, we randomly select the next peer to explore.

```
bash$ ./ion-sampler gnutella --hops 25 -n 10
10.8.65.171:6348
10.199.20.183:5260
10.8.45.103:34717
10.21.0.29:6346
10.32.170.200:6346
10.201.162.49:30274
10.222.183.129:47272
10.245.64.85:6348
10.79.198.44:36520
10.216.54.169:44380
bash$
```

Listing 1: Example usage of the `ion-sampler` tool. We specify that we want to use the Gnutella plug-in, each walk should take 25 hops, and we would like 10 samples. The tool then prints out 10 IP-address:port pairs. We have changed the first octet of each result to "10" for privacy reasons.

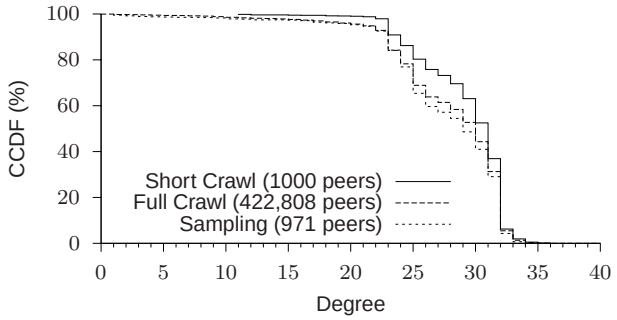


Fig. 8: Comparison of degree distributions observed from sampling versus exhaustively crawling all peers

	Short Crawl	Full Crawl	Sampling
Short Crawl	—	0.120	0.161
Full Crawl	0.120	—	0.043
Sampling	0.161	0.043	—

TABLE III: KS statistic (D) between pairs of empirical datasets

true distribution is $D \leq 0.043$. On the other hand, because the short crawl data *exceeds* the full crawl distribution, its error relative to the true distribution is $D \geq 0.120$. In other words, the true D for the sampling data is *at most* 0.043, while the true D for the short crawl data is *at least* 0.120. It is possible that sampling with MRWB produces more accurate results than a full crawl (which suffers from distortion), but this is difficult to prove conclusively.

C. Efficiency

Having demonstrated the validity of the MRWB technique, we now turn our attention to its efficiency. Performing the walk requires $n \cdot r$ queries, where n is the desired number of samples and r is the length of the walk in hops. If r is too low, significant bias may be introduced. If r is too high, it should not introduce bias, but is less efficient. From graph theory, we expect to require $r \geq O(\log |V|)$ for an ordinary random walk.

To empirically explore the selection of r for Gnutella, we conducted many sets of sampling experiments using different values of r, with full crawls interspersed between the sampling experiments. For each sampling experiment, we compute the KS statistic, D, between the sampled degree distribution and that captured by the most recent crawl. Figure 9 presents the mean and standard deviation of D as a function of r across different experiments. The figure shows that low values of r (≤ 10) can lead to enormous bias ($D \geq 0.4$). The amount of bias decreases rapidly with r, and low bias is observed for $r \geq 25$ hops. However, in a single experiment with $r = 30$ hops, we observed $D > 0.3$, while all other experiments at that length showed $D < 0.09$. Investigating the anomalous dataset, we found that a single peer had been selected 309 out of 999 times.

Further examining the trace of this walk, we found that the walk happened to start at a peer with only a single neighbor. In such a case, the walk gets stuck at that peer due to the way Metropolis–Hastings transitions to a new peer y with probability only $\frac{\text{degree}(x)}{\text{degree}(y)}$. When this "stuck" event occurs late in the walk, it is just part of the normal re-weighting to correct

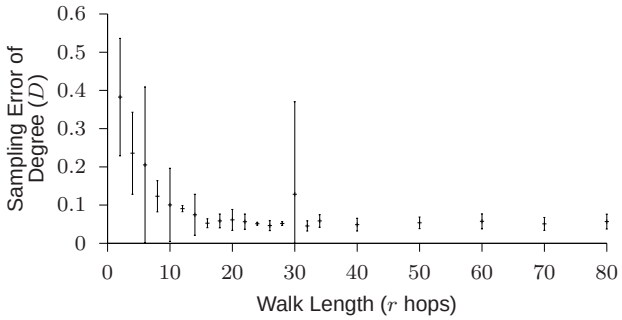


Fig. 9: Difference between sampled results and a crawl as a function of walk length. Each experiment was repeated several times. Error bars show the sample standard deviation.

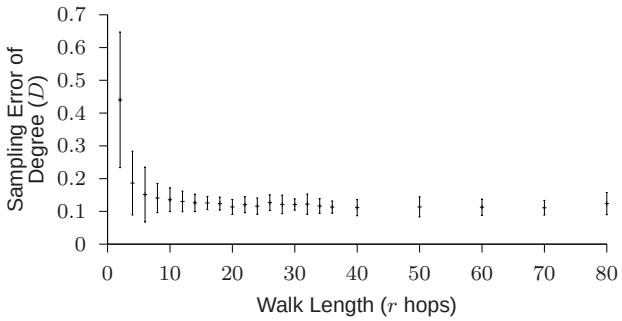


Fig. 10: Difference between sampled results and a crawl as a function of walk length, after the change suggested in Section VI-C. Each experiment was repeated several times. Error bars show the sample standard deviation.

for a regular random walk's bias towards high degree peers. However, when it occurs during the first step of the walk, a large fraction of the walks will end at the unusual low-degree peer, resulting in an anomalous set of selections where the same peer is chosen many times.

One way to address this problem is to increase the walk length by requiring

$$r \geq \frac{\text{maximum degree}}{\text{minimum degree}} \cdot \log |V|.$$

However, this reduces the efficiency of the walk. More importantly, we typically do not accurately know the maximum degree, *i.e.*, while increasing r decreases the probability of an anomalous event, it does not preclude it. Therefore, we suggest the following heuristic to prevent such problems from occurring. During the first few steps of the walk, always transition to the next peer as in a regular random walk; after the first few steps, use the Metropolis–Hastings method for deciding whether to transition to the next peer or remain at the current one. This modification eliminates the correlations induced by sharing a single starting location, while keeping the walk length relatively short. We repeated the experiment after making this change; the results are shown in Figure 10. The observed error in the revised implementation is low for $r \geq 15$, with low variance. In other words, the samples are consistently accurate for $r \geq 15$.

In light of these considerations, we conservatively regard a choice of $r = 25$ as a safe walk length for Gnutella. Choosing $r = 25$, we can collect 1,000 samples by querying 25,000

Fig. 11: Runtime of `ion-sampler` as a function of walk length when collecting 1,000 samples.

peers, over an order of magnitude in savings compared with performing a full crawl which must contact more than 400,000.

D. Execution Time

We examined execution time as a function of the number of hops, r, and present the results in Figure 11. With $r = 25$ hops, the execution time is around 10 minutes. In our initial implementation of `ion-sampler`, a small fraction of walks would get "stuck" in a corner of the network, repeatedly trying to contact a set of departed peers. While the walks eventually recover, this corner-case significantly and needlessly delayed the overall execution time. We added a small cache to remember the addresses of unresponsive peers to address this issue.

For comparison, Cruiser takes around 13 minutes to capture the entire topology. This begs the question: if `ion-sampler` does an order of magnitude less work, why is the running time only slightly better? While `ion-sampler` contacts significantly fewer peers, walks are sequential in nature which limits the amount of parallelism that `ion-sampler` can exploit. Cruiser, on the other hand, can query peers almost entirely in parallel, but it must still do $O(n)$ work, where n is the population size. In other words, if a peer-to-peer network doubles in size, Cruiser will take twice as long to capture it. Alternately, we can keep Cruiser's execution time approximately constant by double the amount of hardware and bandwidth we provision for Cruiser's use. The `ion-sampler` tool requires only $O(logn)$ work, meaning there is little change in its behavior as the network grows.

While longer execution time has a negative impact on the accuracy of Cruiser's results, `ion-sampler`'s results are not significantly impacted by the time required to perform the walk (as demonstrated in Section V-D where we simulate walks of 10,000 hops).

E. Summary

In summary, these empirical results support the conclusion that a Metropolized Random Walk with Backtracking is an appropriate method of collecting measurements from peer-to-peer systems, and demonstrate that it is significantly more accurate than other common sampling techniques. They also illustrate the dramatic improvement in efficiency and scalability of MRWB compared to performing a full crawl. As network size increases, the cost of a full crawl grows linearly and takes longer to complete, introducing greater distortion into the captured snapshots. For MRWB, the cost increases logarithmically, and no additional bias is introduced.

VII. Discussion

A. How many samples are required?

An important consideration when collecting samples is to know how many samples are needed for statistically significant results. This is principally a property of the distribution being sampled. Consider the problem of estimating the underlying frequency f of an event, *e.g.*, that the peer degree takes a particular value. Given N unbiased samples, an unbiased estimate of f is $\widehat{f} = M/N$ where M is the number of samples for which the event occurs. $\widehat{f}$ has root mean square (RMS) relative error

$$\sigma = \sqrt{\mathsf{Var}(\widehat{f})}/f = \sqrt{(1-f)/fN}.$$

From this expression, we derive the following observations:

- Estimation error does not depend on the population size; in particular the estimation properties of unbiased sampling scale independently of the size of the system under study.
- The above expression can be inverted to derive the number of samples $N_{f,\sigma}$ required to estimate an outcome of frequency f up to an error σ. A simple bound is $N_{f,\sigma} \leq 1/(f\sigma^2)$.
- Unsurprisingly, smaller frequency outcomes have a larger relative error. For example, gathering 1,000 unbiased samples gives us very little useful information about events which only occur one time in 10,000; the associated σ value is approximately 3: the likely error dominates the value to be estimated. This motivates using *biased sampling* in circumstances that we discuss in the next subsection.

The presence of sampling bias complicates the picture. If an event with underlying frequency f is actually sampled with frequency f_0, then the RMS relative error acquires an additional term $(1 - f_0/f)^2$ which *does not reduce* as the number of samples N grows. In other words, when sampling from a biased distribution, increasing the number of samples only increases the accuracy with which we estimate the *biased* distribution.

B. Unbiased versus biased sampling

At the beginning of this paper, we set the goal of collecting unbiased samples. However, there are circumstances where unbiased samples are inefficient. For example, while unbiased samples provide accurate information about the body of a distribution, they provide very little information about the tails: the pitfall of estimating rare events we discussed in the previous subsection.

In circumstances such as studying infrequent events, it may be desirable to gather samples with a *known sampling bias*, *i.e.*, with non-uniform sampling probabilities. By deliberately introducing a sampling bias towards the area of interest, more relevant samples can be gathered. During analysis of the data,

each sample is weighted inversely to the probability that it is sampled. This yields unbiased estimates of the quantities of interest, even though the selection of the samples is biased. This approach is known as *importance sampling* [50].

A known bias can be introduced by choosing an appropriate definition of $\mu(x)$ in the Metropolis–Hastings equations presented in Section IV and altering the walk accordingly. Because the desired type of known bias depends on the focus of the research, we cannot exhaustively demonstrate through simulation that Metropolis–Hastings will operate correctly in a dynamic environment for any $\mu(x)$. Our results show that it works well in the common case where unbiased samples are desired (*i.e.*, $\mu(x) = \mu(y)$ for all x and y).

C. Sampling from structured systems

Throughout this paper, we have assumed an unstructured peer-to-peer network. Structured systems (also known as Distributed Hash Tables or DHTs) should work just as well with random walks, provided links are still bidirectional. However, the structure of these systems often allows a more efficient technique.

In a typical DHT scheme, each peer has a randomly generated identifier. Peers form an overlay that actively maintains certain properties such that messages are efficiently routed to the peer "closest" to a target identifier. The exact properties and the definition of "closest" vary, but the theme remains the same. In these systems, to select a peer at random, we may simply generate an identifier uniformly at random and find the peer closest to the identifier. Because peer identifiers are generated uniformly at random, we know they are uncorrelated with any other property. This technique is simple and effective, as long as there is little variation in the amount of identifier space that each peer is responsible for. We made use of this sampling technique in our study of the widely-deployed Kad DHT [51].

VIII. Conclusions and Future Work

This paper explores the problem of sampling representative peer properties in large and dynamic unstructured P2P systems. We show that the topological and temporal properties of P2P systems can lead to significant bias in collected samples. To collect unbiased samples, we present the Metropolized Random Walk with Backtracking (MRWB), a modification of the Metropolis–Hastings technique, which we developed into the `ion-sampler` tool. Using both simulation and empirical evaluation, we show that MRWB can collect approximately unbiased samples of peer properties over a wide range of realistic peer dynamics and topological structures.

We are pursuing this work in the following directions. First, we are exploring improving sampling efficiency for uncommon events (such as in the tail of distributions) by introducing known bias, as discussed in Section VII-B. Second, we are studying the behavior of MRWB under flash-crowd scenarios, where not only the properties of individual peers are changing, but the *distribution* of those properties is also rapidly evolving. Finally, we are developing additional plug-ins for `ion-sampler` and using it in conjunction with other measurement tools to accurately characterize several properties of widely-deployed P2P systems.

Acknowledgments

We would like to thank Amir Rasti and John Capehart for their invaluable efforts in developing the dynamic overlay simulator. We would also like to thank Virginia Lo for her valuable feedback on this paper.

This material is based upon work supported in part by the National Science Foundation (NSF) under Grant No. Nets-NBD-0627202 and an unrestricted gift from Cisco Systems. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF or Cisco.

References

[1] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan, "Chord: A Scalable Peer-to-peer Lookup Protocol for Internet Applications," *IEEE/ACM Transactions on Networking*, 2002.

[2] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A Scalable Content-Addressable Network," in *SIGCOMM*, 2001.

[3] S. Saroiu, P. K. Gummadi, and S. D. Gribble, "Measuring and Analyzing the Characteristics of Napster and Gnutella Hosts," *Multimedia Systems Journal*, vol. 9, no. 2, pp. 170–184, Aug. 2003.

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

[5] D. Stutzbach and R. Rejaie, "Understanding Churn in Peer-to-Peer Networks," in *Internet Measurement Conference*, Rio de Janeiro, Brazil, Oct. 2006.

[6] S. Chib and E. Greenberg, "Understanding the Metropolis–Hastings Algorithm," *The Americian Statistician*, vol. 49, no. 4, pp. 327–335, Nov. 1995.

[7] W. Hastings, "Monte Carlo Sampling Methods Using Markov Chains and Their Applications," *Biometrika*, vol. 57, pp. 97–109, 1970.

[8] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller, "Equations of State Calculations by Fast Computing Machines," *Journal of Chemical Physics*, vol. 21, pp. 1087–1092, 1953.

[9] A. Awan, R. A. Ferreira, S. Jagannathan, and A. Grama, "Distributed Uniform Sampling in Unstructured Peer-to-Peer Networks," in *Hawaii International Conference on System Sciences*, Kauai, HI, Jan. 2006.

[10] Z. Bar-Yossef and M. Gurevich, "Random Sampling from a Search Engine's Index," in *WWW*, 2006.

[11] D. Stutzbach, R. Rejaie, N. Duffield, S. Sen, and W. Willinger, "Sampling Techniques for Large, Dynamic Graphs," in *Global Internet Symposium*, Barcelona, Spain, Apr. 2006.

[12] D. Stutzbach and R. Rejaie, "Capturing Accurate Snapshots of the Gnutella Network," in *Global Internet Symposium*, Miami, FL, Mar. 2005, pp. 127–132.

[13] B. Bollobás, "A probabilistic proof of an asymptotic formula for the number of labelled regular graphs," *European Journal of Combinatorics*, vol. 1, pp. 311–316, 1980.

[14] M. Jerrum and A. Sinclair, "Fast uniform generation of regular graphs," *Theoretical Computer Science*, vol. 73, pp. 91–100, 1990.

[15] C. Cooper, M. Dyer, and C. Greenhill, "Sampling regular graphs and a peer-to-peer network," in *Symposium on Discrete Algorithms*, 2005, pp. 980–988.

[16] V. Krishnamurthy, J. Sun, M. Faloutsos, and S. Tauro, "Sampling Internet Topologies: How Small Can We Go?" in *International Conference on Internet Computing*, Las Vega, NV, June 2003, pp. 577–580.

[17] 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*, Waterloo, Ontario, CA, May 2005.

[18] 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*, vol. 102, no. 12, pp. 4221–4224, Mar. 2005.

[19] A. A. Tsay, W. S. Lovejoy, and D. R. Karger, "Random Sampling in Cut, Flow, and Network Design Problems," *Mathematics of Operations Research*, vol. 24, no. 2, pp. 383–413, Feb. 1999.

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

[21] 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*, Baltimore, MD, May 2005.

[22] 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, pp. 121–128.

[23] M. Henzinger, A. Heydon, M. Mitzenmacher, and M. Najork, "On Near-Uniform URL Sampling," in *International World Wide Web Conference*, May 2001, pp. 295–308.

[24] L. Lovász, "Random walks on graphs: A survey," *Combinatorics: Paul Erdös is Eighty*, vol. 2, pp. 1–46, 1993.

[25] 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.

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

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

[28] V. Vishnumurthy and P. Francis, "On Heterogeneous Overlay Construction and Random Node Selection in Unstructured P2P Networks," in *INFOCOM*, Barcelona, Spain, Apr. 2006.

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

[30] D. Heckathorn, "Respondent-driven sampling: a new approach to the study of hidden populations," *Social Problems*, vol. 44, no. 2, pp. 174–199, 1997.

[31] L. Goodman, "Snowball Sampling," *Annals of Math. Statistics*, vol. 32, pp. 148–170, 1961.

[32] E. Deaux and J. Callaghan, "Key informant versus self-report estimates of health behavior," *Evaluation Reviews*, vol. 9, pp. 365–368, 1985.

[33] J. Watters and P. Biernack, "Targeted sampling: Options for the study of hidden populations," *Annual Review of Sociolog*, vol. 12, pp. 401–429, 1989.

[34] M. Salganik, "Variance estimation, design effects, and sample size calculation for respondent-driven sampling," *Journal of Urban Health (to appear)*, 2006.

[35] J. Leskovec, J. Kleinberg, and C. Faloutsos, "Graphs over Time: Densification Laws, Shrinking Diameters and Possible Explanations," in *KDD*, Chicago, IL, Aug. 2005.

[36] A. Bonato, "A survey of models of the web graph," in *Combinatorial and Algorithmic Aspects of Networking*, 2004, pp. 159–172.

[37] 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)*, Ithaca, NY, Feb. 2005.

[38] 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*, Apr. 2004.

[39] D. Stutzbach, R. Rejaie, and S. Sen, "Characterizing Unstructured Overlay Topologies in Modern P2P File-Sharing Systems," in *Internet Measurement Conference*, Berkeley, CA, Oct. 2005, pp. 49–62.

[40] K. P. Gummadi, S. Saroiu, and S. D. Gribble, "King: Estimating Latency between Arbitrary Internet End Hosts," in *Internet Measurement Workshop*, Marseille, France, Nov. 2002.

[41] D. Liben-Nowell, H. Balakrishnan, and D. Karger, "Analysis of the Evolution of Peer-to-Peer Systems," in *Principles of Distributed Computing*, Monterey, CA, July 2002.

[42] S. Rhea, D. Geels, and J. Kubiatowicz, "Handling Churn in a DHT," in *USENIX*, 2004, pp. 127–140.

[43] J. Li, J. Stribling, F. Kaashoek, R. Morris, and T. Gil, "A Performance vs. Cost Framework for Evaluating DHT Design Tradeoffs under Churn," in *INFOCOM*, Miami, FL, Mar. 2005.

[44] F. E. Bustamante and Y. Qiao, "Friendships that last: Peer lifespan and its role in P2P protocols," in *International Workshop on Web Content Caching and Distribution*, 2003.

[45] S. Sen and J. Wang, "Analyzing Peer-To-Peer Traffic Across Large Networks," *IEEE/ACM Transactions on Networking*, vol. 12, no. 2, pp. 219–232, Apr. 2004.

[46] K. P. Gummadi, R. J. Dunn, S. Saroiu, S. D. Gribble, H. M. Levy, and J. Zahorjan, "Measurement, Modeling, and Analysis of a Peer-to-Peer File-Sharing Workload," in *SOSP*, 2003.

[47] D. Leonard, V. Rai, and D. Loguinov, "On Lifetime-Based Node Failure and Stochastic Resilience of Decentralized Peer-to-Peer Networks," in *SIGMETRICS*, 2005.

[48] H. Dämpfling, "Gnutella Web Caching System: Version 2 Specifications Client Developers' Guide," http://www.gnucleus.com/gwebcache/newgwc.html, June 2003.

[49] P. Karbhari, M. Ammar, A. Dhamdhere, H. Raj, G. Riley, and E. Zegura, "Bootstrapping in Gnutella: A Measurement Study," in *PAM*, Apr. 2004.

[50] R. Srinivasan. Berlin, Germany: Springer-Verlag, 2002.

[51] D. Stutzbach and R. Rejaie, "Improving Lookup Performance over a Widely-Deployed DHT," in *INFOCOM*, Barcelona, Spain, Apr. 2006.

PLACE PHOTO HERE

Daniel Stutzbach (M'06) is the president of Stutzbach Enterprises, LLC. He completed his Ph.D. at the University of Oregon in 2006, where he focused on measurement and modeling of peer-to-peer systems. Prior to joining the UO, Daniel worked as an embedded systems programmer, developing firmware for routers. Daniel earned his B.S. degree from Worcester Polytechnic Institute in 1998. Daniel has been a member of the ACM since 2003 and the IEEE since 2005.

PLACE PHOTO HERE

Reza Rejaie (SM'06) is currently an Assistant Professor at the University of Oregon. From 1999 to 2002, he was a Senior Technical Staff member at AT&T Labs—Research in Menlo Park, California. He received a NSF CAREER Award for his work on P2P streaming in 2005. Reza received his M.S. and Ph.D. degrees from the University of Southern California in 1996 and 1999, and his B.S. degree from the Sharif University of Technology in 1991. Reza is a senior member of both the ACM and IEEE.

PLACE PHOTO HERE

Nick Duffield (F'05) is a Distinguished Member of Technical Staff and an AT&T Fellow in the Internet & Network Systems Research Laboratory at AT&T Labs—Research, where he has been since 1995. He previously held postdoctoral and faculty positions in Dublin, Ireland and Heidelberg, Germany. He received a Ph.D. from the University of London, UK, in 1987. His current research focuses on measurement and inference of network traffic. He was charter Chair of the IETF working group on Packet Sampling, and co-inventor of the Smart Sampling technologies that lie at the heart of AT&T's scalable Traffic Analysis Service. He is a Fellow of the IEEE. He is an Associate Editor of the IEEE/ACM Transactions on Networking.

PLACE PHOTO HERE

Subhabrata Sen (M'01) received a Bachelor of Engineering (First Class with Honors) degree in Computer Science (1992) from Jadavpur University, India, and M.S. and Ph.D. degrees in Computer Science from the University of Massachusetts, Amherst, USA, in 1997 and 2001, respectively. Dr. Sen is currently a Principal Member of Technical Staff in the Internet & Network Systems Research Laboratory at AT&T Labs—Research, where he has been since 2001. His research interests include IP network management, traffic analysis, network data mining, security and anomaly detection, peer-peer systems, and end-to-end support for streaming multimedia. He is a member of the ACM.

PLACE PHOTO HERE

Walter Willinger (F'95) received the Diplom (Dipl. Math.) from the ETH Zurich, Switzerland, and the M.S. and Ph.D. degrees from the School of ORIE, Cornell University. He is a member of the Information and Software Systems Research Center at AT&T Labs–Research. Previously, he was a Member of Technical Staff at Bellcore Applied Research (1986–1996). He is a Fellow of ACM (2005) and a Fellow of IEEE (2005). For his work on the self-similar ("fractal") nature of Internet traffic, he received the 1996 IEEE W.R.G. Baker Prize Award, the 1994 W.R. Bennett Prize Paper Award, and the 2005 ACM/SIGCOMM "Test of Time" Paper Award. He is an AT&T Fellow (2007) and a Fellow of ACM (1995).