Random Walk Based Node Sampling in Self-Organizing Networks

Ming Zhong
Department of Computer Science
University of Rochester
zhong@cs.rochester.edu

Kai Shen
Department of Computer Science
University of Rochester
kshen@cs.rochester.edu

ABSTRACT

Random walk is a means of network node sampling that requires little index maintenance and can function on almost all connected network topologies. With careful guidance, node samples following a desired probability distribution can be generated with the only requirement that the sampling probabilities of each visited node and its direct neighbors are known at each walk step. This paper describes a broad range of network applications that can benefit from such guided random walks in dynamic and decentralized settings. This paper also examines several key issues for implementing random walks in self-organizing networks, including the convergence time of random walks, impact of dynamic network changes and particularly resulted walker losses, and the difficulty of pacing walk steps without synchronized clocks between network nodes. Our result suggests that with proper management, these issues do not cause significant problems under many realistic network environments.

1. INTRODUCTION

Random walk is a way to sample network nodes — at each step, a walker randomly chooses its next hop (among direct neighbors of the current node) following certain probabilistic preference for each neighbor. With careful guidance at each step of the walk [12, 21], one can produce node samples following a desired probability distribution. Such a tunable probabilistic node sampling is a fundamental building block for many network applications, such as non-uniform membership management [30], load balancing [13], small-world routing [16], peer-to-peer (p2p) search [29], and biased information propagation [15].

Random walk is particularly attractive to self-organizing networks like Internet overlay networks and wireless ad hoc networks. In these systems, nodes can join and leave dynamically without centralized control, and the network topology itself can also change over time. Random walk requires little index or state maintenance and it can function on almost all connected network topologies. In these aspects, it is superior to systems with sophisticated index states or rigid network structures, *e.g.*, distributed hash tables (DHTs) [25, 26, 28]. Compared with index-free node traversal schemes like network flooding, random walk is inherently scalable in that its network communication overhead does not increase as the network size grows. Furthermore, guided random walk is very flexible in achieving probabilistic (probably biased) node sampling.

Despite its potential advantages, several issues need to be addressed concerning random walks in self-organizing networks. First, random walk based node sampling can take significant warm-up time to converge to the desired sampling distribution. Such convergence is dependent on the sampling distribution, network diameter, and the expansion property of network topologies. Second, random walks may be affected by dynamic network changes. In particular, a walker may be lost due to link changes or node departures in self-organizing networks. Third, without synchronized clocks at network nodes, it is hard to guarantee equal time interval for each walk step due to variable network link latencies. We examine these issues in this paper. Our goal is to provide a high-level understanding on the effectiveness and suitability of using guided random walks in self-organizing network applications.

A large body of previous works have already employed random walks in network applications. Examples include random walk based search [1, 8, 11, 20], network topology construction [10, 18, 19], and information gathering [3]. These studies focus on the functionality and performance of random walks in supporting specific network management functions and applications. They do not address general issues concerning random walks in self-organizing networks that we study in this paper. More specifically, previous studies provided little result on random walk convergence, walker loss, or the pacing of walkers in self-organizing networks.

The rest of this paper is organized as follows. Section 1.1 describes the background of our random walk based probabilistic network node sampling. Section 2 illustrates a broad set of network applications that can benefit from random walk based node sampling. Sections 3, 4, and 5 study three specific issues for random walks in self-organizing networks. Section 6 concludes the paper with a summary of our findings.

1.1 Background

Our random walks build on the Metropolis-Hastings algorithm [12, 21] — an approach to assign transition probabilities to Monte Carlo Markov chains so that they converge to specified probability distributions.

Starting from an initial node, a random walk travels over the network step by step. At each step, the random walk either stays at the current node or moves to one of its direct neighbors in a probabilistic fashion. Specifically, the

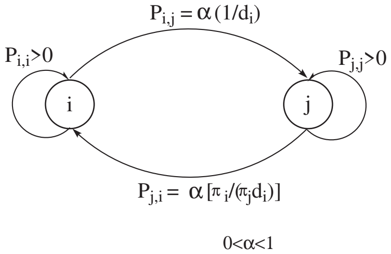

Figure 1: The transition probabilities between i, j when $\frac{\pi_i}{d_i} \leq \frac{\pi_j}{d_j}$ (without loss of generality). d_i and d_j are the degrees of nodes i and j respectively.

random walk goes from node i to node j with probability $P_{i,j}$ and $P_{i,j} = 0$ for non-adjacent nodes i, j. The probability for a walker at node i to stay unmoved at a step $P_{i,i} = 1 - \sum_{j \in \ neighbors(i)} P_{i,j}$. A random walk achieves convergence if its node visitation probability distribution remains unchanged (or changes very slightly) after any number of future steps.

The Metropolis-Hastings algorithm guides the configuration of random walks such that they converge to the application-desired distribution, π with π_i be the desired probability of visiting node i. For any pair of adjacent nodes i, j, Figure 1 provides the guidance for setting appropriate transition probabilities when $\frac{\pi_i}{d_i} \leq \frac{\pi_j}{d_j}$ (without loss of generality). Intuitively, Figure 1 achieves the desired convergence to π because there are always balanced probability flows between arbitrary i, j once the random walk starts to visit i, j with probabilities π_i, π_j respectively. In particular, a laziness factor $0 < \alpha < 1$ is used to discount probability flows, which ensures that each step of the random walk has non-zero probability to stay at the current node. Such self-loop enabled random walks are guaranteed to have unique converged distribution on connected topologies [9, 24]. It is worth noting that the movement of random walks can be determined solely based on the local knowledge about the current node and its direct neighbors.

Let us consider a special case of probabilistic node sampling — uniform node sampling, *i.e.*, where $\forall i, j, \pi_i = \pi_j$. According to Figure 1, the uniform node sampling can be easily achieved by setting

$$P_{i,j} = \begin{cases} \alpha \cdot \frac{1}{d_i} & \text{if } d_i \geq d_j; \\ \alpha \cdot \frac{1}{d_j} & \text{if } d_i < d_j. \end{cases} \qquad (0 < \alpha < 1)$$

where i, j are adjacent nodes.

2. APPLICATIONS OF RANDOM WALKS

Probabilistic node sampling is essential for many network applications. Our guided random walks can naturally benefit these applications. Below we describe several examples.

Random membership subset

In a random membership subset service, each node maintains a small, dynamically changing, random membership subset with uniform representation of network members rather than a full member list. The scalability is achieved in that per-node random membership subsets can be much smaller than the complete list. For many applications, making decisions based on random membership subsets has comparable performance with knowing the complete network membership. Kostić *et al.* proposed a random membership subset service for tree-shaped network topologies [17] but it cannot be easily applied to more general mesh-like network structures. In comparison, random walk based node sampling can function on almost any connected network topologies. Furthermore, it can also support non-uniform membership management as follows [30]. Each node i initiates k_i independent random walks $R_{i,1}, R_{i,2}, ..., R_{i,k_i}$ where k_i is the size of node i's membership subset. Guided by the Metropolis-Hastings algorithm described in Section 1.1, the random walks are configured such that they all converge to the desired sampling distribution over all network nodes. Whenever node j receives a random walk $R_{i,l}$ initiated from node i, node j sends its own identity to node i. Upon receiving j's identity referred by $R_{i,l}$, node i updates the lth member of its membership subset with j.

Load balancing

A key issue for dynamic load balancing in self-organizing networks is to identify *hotspots* (highly overloaded nodes) and reassign their loads to others. In this application, each load balancing node tries to learn the load information on some other network nodes. When a large load variation exists between the local node and a remote node, a load transfer session is initiated to balance the load. Probabilistic node sampling with load-biased probability distributions (*e.g.*, choose a node with probability proportional to its load) is more likely to find hotspots than uniform sampling used in some current p2p load balancing algorithms [13]. This is especially the case when there is a small number of highly loaded nodes in the network (*e.g.*, when the load distribution is powerlaw).

Small-world routing

Simple greedy routing in a two-dimensional grid takes $O(\sqrt{n})$ hops, where n is the total number of nodes in the network. Kleinberg [16] illustrates a small-world routing as follows. By having each node connect to one remote node (chosen randomly following a distance-dependent probabilistic distribution), it is possible to reduce the greedy routing time to $O(\log^2 n)$ hops. Random walk based node sampling is instrumental for easy creation and maintenance of the random remote link at each node.

Peer-to-peer search

The *square-root principle* [7] is known to achieve low search time for p2p search techniques that do not utilize query routing indices. Under this principle, each object is probed with probability proportional to the square root of its query popularity. Earlier search methods realize the square-root principle by using either object replication [7] or topology reconstruction [8], which may not be desirable for those applications with large, dynamic datasets and limited network bandwidth. Being biased towards peers with popular contents, random walk based probabilistic node sampling can achieve the same goal without the need of object transfer or network topology changes [29]. For each node probed by the random walk, local matched objects, if discovered, are returned to the query initiating node.

Applications	Time norm.	Callback
Random membership subset	Yes	Yes
Load balancing	Yes	Yes
Small-world routing	Yes	Yes
Peer-to-peer search	No	Maybe
Information propagation	No	No

Table 1: Application properties on 1) time normalization; and 2) whether a callback to the original node is needed at each step of the walk.

Distance-biased information propagation

Gossip-based broadcast algorithms [14, 23] provide a robust and scalable mechanism for distributed information dissemination. In many network applications (*e.g.*, failure detection), new information is more "interesting" to nodes that are nearby. Kempe *et al.* [15] show that with a carefully chosen probability distribution, their non-uniform gossip algorithms can provide both short broadcast time and distance-biased propagation time bounds (information tend to reach nearby nodes first). Their algorithm assumes the existence of a scalable mechanism to identify random nodes with a given distance-biased probability distribution. Such probabilistic node sampling can be easily achieved by our guided random walks.

Application Properties

Although all the above mentioned applications employ probabilistic node sampling as a fundamental building block, they differ in various ways. We are specifically interested in two application properties that affect the management of random walks.

- For some applications, node sampling probabilities concern the chance for the node to be sampled *at a given step of the random walk*. For other applications, however, the sampling probability concerns the chance for the node to be sampled *at a given time*. For the latter case (we call *time-normalized* probabilistic node sampling), the variation of the interval time at each walk step may affect the node sampling probability. More specifically, walk steps with longer per-step interval time would result in larger time-normalized sampling probabilities for the corresponding nodes.
- For some applications, the random walker calls back its original node after visiting each new node. Such callback is needed for applications maintaining the set of sampled nodes at the original node. For object search applications, callback might or might not be needed at each step of the walk depending on whether the visited node has any query matches. Walker callbacks can be utilized to piggyback additional information desired by the management of random walks.

Table 1 summarizes these two properties for our example applications. Their detailed effects on random walk management will be discussed in later sections.

3. RANDOM WALK CONVERGENCE

The Metropolis-Hastings algorithm described in Section 1.1 guarantees that an appropriately configured random walk converges to the desired node sampling distribution. However, random walks often take non-negligible convergence time in typical self-organizing networks. For instance, after one step of the walk, the walker cannot go beyond the immediate neighbors of the starting node no matter how intelligent the walker is. In this section, we study the convergence time of random walk based probabilistic node sampling.

Network topologies

For a specific random walk configuration, its convergence time is determined by the network topology, and particularly the network diameter and its expansion property. 1) Random walks tend to converge faster on networks with smaller diameters. This is because random walks can quickly reach remote nodes in these networks. 2) For a network $G = (V, E)$ with $|V| = n$, its expansion property is defined as

$$h(G) = \min_{S \subset V \wedge 1 \leq |S| \leq \frac{n}{2}} \frac{|\vartheta(S)|}{|S|} \tag{1}$$

where $\vartheta(S)$ is the set of edges with exactly one endpoint in S. Intuitively, the expansion property characterizes the bottleneck for traveling over the topology. When the expansion is low, random walks may get trapped in a small network area, which slows down their convergence to desired sampling probabilities.

We examine the network diameter and expansion property of several common self-organizing network topologies:

- *Two-dimensional (2-d) tori.* Tori-like structures have been used for self-organizing content addressable networks [25]. Structures like 2-d tori are also common in geographically-constrained networks (where nodes' transmission ranges are limited by geographical distances) such as wireless ad hoc networks.
- *Chord topologies* [28]. An n-node ring-like network with each node also connected to (besides its direct neighbors on the ring) its 2-hop neighbors, 4-hop neighbors, ..., $\frac{n}{2}$-hop neighbors on the ring.
- *Random powerlaw topologies.* In a random powerlaw network topology, each node is connected to a constant number of nodes chosen randomly with probability proportional to their degrees. Such a preferential link creation process generates networks with power-law degree distributions [2, 6].
- *Random regular topologies.* Random regular topologies model those network applications in which each node is connected to some random nodes and node degrees are upper-bounded by a predetermined threshold [27].

Table 2 shows the network diameters and expansion properties of these network topologies. First, 2-d tori have the largest values and hence it has the slowest convergence. Other three topologies are known to possess low diameters and high expansion properties (either constant or logarithmic in the network size), which leads to quick convergence.

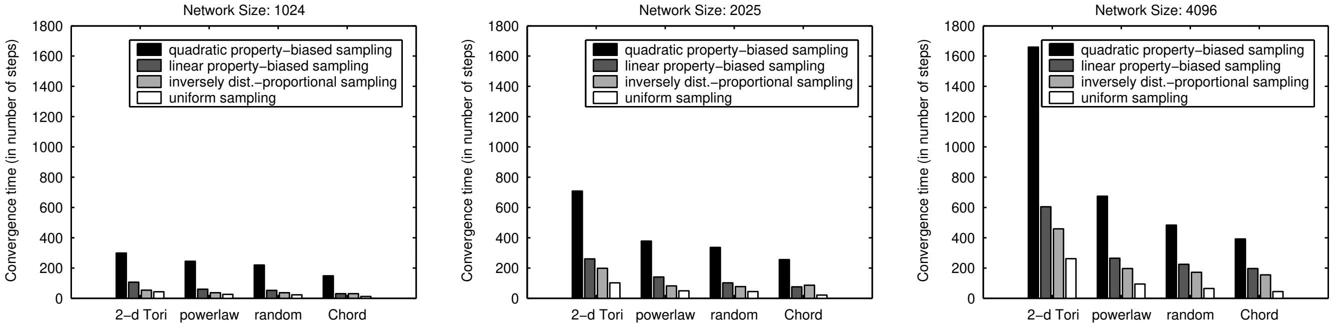

Figure 2: Random walk convergence time. Here *inversely distance-proportional sampling* chooses a node with probability inversely proportional to its distance from the random walk originating node. *Linear property-biased sampling* selects a node with probability proportional to its property value. *Quadratic property-biased sampling* samples a node with probability proportional to the square of its property value.

	2-d tori	Chord	random powerlaw	random regular
Diameter	$\sqrt{n}$	$\log n$ [19]	$O(\log n)$ w.h.p. [5]	$O(\log n)$ w.h.p [4]
Expansion	$O(\frac{1}{\sqrt{n}})$	$O(\frac{1}{\sqrt{\log n}})$ [19]	$\Omega(1)$ w.h.p. [22]	$\Omega(1)$ w.h.p [4]

Table 2: The diameters and expansion properties of several common network topologies. n is the number of nodes in the network.

Node sampling distributions

In addition to the network topology, the desired sampling distribution also affects the random walk convergence time in that edges with small transition probabilities may slow down random walks. The problem may become more severe when these edges happen to be the bottlenecks for random walk movements as revealed by topology expansion. In principle, the random walk converges slower to more skewed node sampling distributions. Here we list some typical sampling distributions employed by network applications:

- *Uniform random sampling* — each node has an equal chance of being sampled. This is the most basic random node sampling in which no sampling bias is needed.
- *Distance-biased sampling.* Small-world routing or information propagation often need to sample nodes with probabilities dependent on its distance from the original node, *e.g.*, inversely proportional to the distance.
- *Property-biased sampling* — sampling based on per-node capacities, loads, or content qualities. Many applications, including load balancing and search, benefit from quickly discovering nodes with high loads, capacities, or content qualities. This can be achieved by using a variety of biased sampling distributions with different bias parameters, *e.g.*, sampling nodes with probability proportional to their property values.

Simulation Results

We provide simulation results to quantitatively assess the convergence time under common network topologies and node sampling distributions.

We define the convergence time of a random walk as the number of walk steps needed before the walker gets close enough to the desired node sampling distribution. We provide a formal definition for the convergence time preceded by a definition of the difference metric between two probability distributions.

DEFINITION 1. *The* difference *between two arbitrary probability distributions, x and y, is defined as $\|x, y\| = \frac{1}{2}\sum_i |x_i - y_i|$. The factor $\frac{1}{2}$ is to ensure that the maximum difference does not exceed 1.*

DEFINITION 2. *Let $\pi^{(t)}$ denote the node visitation distribution of a random walk exactly after t steps. Let π be the desired sampling distribution of the random walk. The convergence time (in number of steps) is defined as* $\min\{t : \forall t' \geq t, \|\pi^{(t')}, \pi\| < 0.01\}$.

In our simulations, the average node degrees for random topologies and random powerlaw topologies are set at 8. Figure 2 presents the random walk convergence time under different network sizes, topologies, and sampling distributions. The results show that the convergence time of different network topologies follows the order of “2-d tori” $>$ “random powerlaw” $>$ “random regular” $>$ “Chord”. This mostly matches the order of their network diameters and expansion properties shown in Table 2. However, it is worth noting that Chord networks unexpectedly outperform random powerlaw and random regular topologies. This is because the average node degree of Chord topologies ($\log n$) is higher than that of these random topologies (a constant). More per-node links increase expansion properties and reduce diameters.

Figure 2 also shows that the convergence time of different node sampling distributions. For property-biased sampling in our simulation, per-node properties follow Zipf’s distribution — the number of nodes that have their capacity/load/quality values equal to k is proportional to $\frac{1}{k}$. Results suggest that the order of convergence time follows “quadratic property-biased sampling” $>$ “linear property-biased sampling” $>$ “inversely distance-proportional sampling" > "uniform random sampling". This result is consistent with the order of the skewness in their sampling distributions. The convergence is particularly fast with uniform sampling, *e.g.*, less than 300 steps on 2-d tori and less than 100 steps on other studied topologies are needed to achieve convergence on 4096-node networks. These numbers are not significantly higher than the network diameters of respective topologies — absolute lower-bounds for the convergence time.

4. IMPACT OF NETWORK CHANGES

Convergence in self-organizing networks must consider dynamic network changes, such as node arrivals, node departures, and network topology changes. These network changes do not cause substantial convergence problems because our guided random walks only require a limited amount of local knowledge — the degree and sampling probabilities of the direct neighbors of each node. More specifically, our random walks can accommodate dynamic network changes as long as the required state at each node (information concerning direct neighbors only) can be properly updated. However, one problem due to dynamic network changes warrants further attention — a walker may be lost due to link adjustment or node departure in self-organizing networks. To maintain correct node sampling, walker losses must be promptly discovered and new walkers must be initiated.

Walker loss

We describe two strategies to recover from walk losses:

- *Periodic callbacks.* Each walker makes periodic callbacks to the originating node. If a sufficient number of callbacks are not received in a row, the walker is considered lost and a new walker will be initiated.
- *Reincarnation at TTL.* Each random walker is associated with a certain Time-To-Live (TTL). The walker will stop propagating after TTL number of walk steps. The walker originating node keeps a timer that expires after the TTL walk steps worth of time has elapsed. A new walker will be initiated at such time. If the walker is lost before the TTL expires, its replacement is not initiated until the expiration time.

The main weakness with periodic callbacks lies in the overhead of callbacks. The overhead is cheaper for networks in which it takes a single network hop to communicate with the walker originating node (*e.g.*, Internet overlay networks) while it is more expensive for other networks (*e.g.*, wireless ad hoc networks). On the other hand, some random walk based applications require callback messages as part of the application semantics (described in Section 2). For these applications, periodic walker callbacks for loss detection can be piggybacked in application callback messages and consequently they are almost free.

Unlike periodic callbacks, reincarnation at TTL requires no additional network overhead. However, its recovery of walker losses may not be prompt — if a walker is lost soon after it leaves the originating node, a replacement walker will not be initiated until the full TTL timer expires. At the other end, reincarnation at TTL forces walker re-initiation even if the previous walker has not been lost, thus requiring additional random walk convergence time to reach desired node sampling distribution. An additional problem with this scheme is that it is hard to set the reincarnation timer if the time for each walk step is unpredictable. This is possible in networks where nodes do not have synchronized clocks and network latencies between nodes are unknown.

5. PACING OF WALK STEPS

We define *walk step pacing* as the condition under which each walk step takes an equal time interval. Pacing is important for applications requiring time-normalized probabilistic node sampling (described in Section 2) such as random membership subset, load balancing, and small-world routing. With unequal walk step interval time, the step-wise node sampling distribution supported by the Metropolis-Hastings algorithm is not equal to the time-normalized sampling distribution. Additionally, pacing makes it easy to determine when a walker finishes a certain number of steps and thus it is straightforward to set the TTL expiration timer if reincarnation at TTL is used to recover from walker losses.

Walk step pacing is usually realized by resting the walker at each visited node for a certain amount of time. Overall, the walk step time interval includes the rest time and the network delay as well. Walk step pacing may not be ensured if network latencies between nodes vary significantly and nodes in the network do not have synchronized clocks (which is common in many self-organizing networks). In these cases, additional network communications are needed to realize strict pacing. In particular, a walker can periodically contact its originating node to synchronize and adjust its walk speed. These communications can be piggybacked with other messages when some form of walker callbacks are already employed (as part of application semantics or for walker loss detection).

Even if strict pacing cannot be realized, we can still bound the difference between the step-wise node sampling distribution and the time-normalized distribution if the variations among walk step time intervals are small. This is usually the case when we can keep the walker rest time at each step to be a constant and the rest time is significantly larger than possible network link latency variations.

Let the average walk step time interval when the walker passes node i be t_i and let the average time interval over all walk steps be $\overline{t}$. Let π be the converged step-wise node sampling distribution and let $\pi^{\mathcal{T}}$ be the converged time-normalized node sampling distribution. Then for any node i, we have $\frac{\pi_i^{\mathcal{T}}}{\pi_i} = \frac{t_i}{\overline{t}}$. We assume the variations among walk step time intervals are small. In other words, we have $|t_i - \overline{t}| \leq \alpha \cdot \overline{t}$ for all node i, where $0 < \alpha \ll 1$. Then we have

$$\begin{aligned} \|\pi^{\mathcal{T}}, \pi\| &= \frac{1}{2} \sum_i |\pi_i^{\mathcal{T}} - \pi_i| \\ &\leq \frac{1}{2} \sum_i \alpha \cdot \pi_i \qquad (2) \\ &= \frac{1}{2} \alpha \end{aligned}$$

where the difference between two distributions $\|x, y\|$ was defined in Definition 1 at Section 3.

We consider a typical Internet overlay network environment in which the per-link latency is bounded by 200 ms. If the average random walk step takes 10 second, then $\alpha = 0.02$ so $\|\pi^{\mathcal{T}}, \pi\| \leq 0.01$. In this case, the difference between the stepwise node sampling distribution and the time-normalized sampling distribution is very small even at the absence of any specific mechanism supporting walk step pacing.

6. CONCLUSION

This paper highlights the effectiveness and challenges in using guided random walks to support applications in self-organizing networks, including membership management, load balancing, search, routing, and information propagation. We make the case for the general feasibility of random walks in achieving application-specific probabilistic node sampling. We also examines three important issues that concern the random walks in practical self-organizing network environments: random walk convergence, walker loss, and the pacing of walkers. Our results are summarized below.

First, random walks tend to converge faster when the sampling distributions are less skewed and the underlying topologies have small diameters and high expansions. Careful quantitative evaluations may be necessary to achieve desired application goals without excessive convergence time. For instance, our results suggest a trade-off between sampling bias parameters and the convergence time — a more biased node sampling is more effective in discovering nodes with high capacities, loads, or content qualities but it requires longer warm-up time.

Second, we examined two strategies to address possible walker losses: periodic callbacks and reincarnation at TTL. Our basic conclusion is that periodic callbacks should always be used when the callback overhead is acceptable (*e.g.*, when callbacks can be piggybacked in application messages or when one hop callback is feasible as in Internet overlay networks). When reincarnation at TTL has to be employed, the TTL setting must be chosen with care. It needs to be large enough so that the random walker would not reach the TTL until long after it converges. It also needs to be small enough so that it would not wait too long to recover from a walker loss.

Third, achieving strict walker spacing on network nodes without synchronized clocks requires additional communication overhead. However, even without strict walker spacing, the deviation between step-wide node sampling distribution and the time-normalized sampling distribution is small under many realistic network environments.

7. REFERENCES

[1] L. Adamic, B. Huberman, R. Lukose, and A. Puniyani. Search in Power Law Networks. *Physical Review*, (64):46135–46143, 2001.

[2] A. Barabási and R. Albert. Emergence of Scaling in Random Networks. *Science*, 286:509–512, 1999.

[3] A.R. Bharambe, M. Agrawal, and S. Seshan. Mercury: supporting scalable multi-attribute range queries. In *Proc. of the ACM SIGCOMM*, pages 353–366, Portland, OR, August 2004.

[4] B. Bollobás. *Random Graphs*. Academic Press, London, UK, 1985.

[5] B. Bollobás and O. Riordan. The Diameter of a Scale-free Random Graph. *Combinatorica*, 24(1):5–34, 2004.

[6] B. Bollobás, O. Riordan, J. Spencer, and G. Tusnady. The Degree Sequence of a Scale-free Random Graph Process. *Random Structures and Algorithms*, 18(3):279–290, 2001.

[7] E. Cohen and S. Shenker. Replication Strategies in Unstructured Peer-to-Peer Networks. In *Proc. of the ACM SIGCOMM*, Pittsburgh, PA, August 2002.

[8] B.F. Cooper. Quickly Routing Searches Without Having to Move Content. In *Proc. of the 4th International Workshop on Peer-to-Peer Systems (IPTPS)*, Ithaca, NY, February 2005.

[9] W. Doeblin. Exposé de la théorie des chaînes simples constantes de Markov á un nombre fini d'états. *Mathématique de l'Union Interbalkanique*, 2:77–105, 1938.

[10] C. Gkantsidis, M. Mihail, and A. Saberi. Random Walks in Peer-to-peer Networks. In *Proc. of the IEEE INFOCOM*, pages 120–130, Hong Kong, China, March 2004.

[11] C. Gkantsidis, M. Mihail, and A. Saberi. Hybrid Search Schemes for Unstructured Peer-to-peer Networks. In *Proc. of the IEEE INFOCOM*, pages 1526–1537, Miami, FL, March 2005.

[12] W.K. Hastings. Monte Carlo Sampling Methods Using Markov Chains and Their Applications. *Biometrika*, 57:97–109, 1970.

[13] D. Karger and M. Ruhl. Simple Efficient Load Balancing Algorithms for Peer-to-Peer Systems. In *Proc. of the 16th ACM Symp. on Parallelism in Algorithms and Architectures (SPAA)*, pages 36–43, Barcelona, Spain, June 2004.

[14] R. Karp, C. Schindelhauer, S. Shenker, and B. Vöcking. Randomized Rumor Spreading. In *Proc. of the 41st IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 565–574, Redondo Beach, CA, November 2000.

[15] D. Kempe, J. Kleinberg, and A. Demers. Spatial Gossip and Resource Location Protocols. In *Proc. of the 33rd ACM Symp. on Theory of computing (STOC)*, pages 163–172, Hersonissos, Greece, July 2001.

[16] J. Kleinberg. The Small-World Phenomenon: An Algorithm Perspective. In *Proc. of the 32nd ACM symp. on Theory of Computing (STOC)*, pages 163–170, Portland, OR, 2000.

[17] D. Kostić, A. Rodriguez, J. Albrecht, A. Bhirud, and A. Vahdat. Using Random Subsets to Build Scalable Network Services. In *Proc. of the 4th USENIX Symp. on Internet Technologies and Systems (USITS)*, Seattle, WA, March 2003.

[18] C. Law and K. Siu. Distributed Construction of Random Expander Networks. In *Proc. of the IEEE INFOCOM*, pages 2133–2143, San Francisco, CA, March 2003.

[19] D. Loguinov, A. Kumar, V. Rai, and S. Ganesh. Graph-Theoretic Analysis of Structured Peer-to-Peer Systems: Routing Distances and Fault Resilience. In *Proc. of the ACM SIGCOMM*, pages 395–406, Karlsruhe, Germany, August 2003.

[20] Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker. Search and Replication in Unstructured Peer-to-Peer Networks. In *Proc. of the ACM International Conference on Supercomputing (ICS)*, pages 84–95, New York, NY, June 2002.

[21] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, and E. Teller. Equation of State Calculations by Fast Computing Machines. *Journal of Chemical Physics*, 21:1087–1092, 1953.

[22] M. Mihail, C. Papadimitriou, and A. Saberi. On Certain Connectivity Properties of the Internet Topology. In *Proc. of the 44th IEEE Symposium on Foundations of Computer Science (FOCS)*, page 28, 2003.

[23] B. Pittel. On Spreading A Rumor. *SIAM J. Applied Math.*, 47(1):213–223, February 1987.

[24] D. Randall. Mixing. A Tutorial in the IEEE FOCS, 2003.

[25] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A Scalable Content-Addressable Network. In *Proc. of the ACM SIGCOMM*, pages 161–172, San Diego, CA, August 2001.

[26] A. Rowstron and P. Druschel. Pastry: Scalable, Distributed Object Location and Routing for Large-scale Peer-to-Peer Systems. In *Proc. of IFIP/ACM Middleware Conference*, pages 329–350, Heidelberg, Germany, November 2001.

[27] K. Shen. Structure Management for Scalable Overlay Service Construction. In *Proc. of the First USENIX/ACM Symp. on Networked Systems Design and Implementation (NSDI)*, pages 281–294, San Francisco, CA, March 2004.

[28] I. Stoica, R. Morris, D. Karger, M.F. Kaashoek, and H. Balakrishnan. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. In *Proc. of the ACM SIGCOMM*, pages 149–160, San Diego, CA, August 2001.

[29] M. Zhong and K. Shen. Popularity-Biased Random Walks for Peer-to-Peer Search Under the Square-Root Principle. In *Proc. of the 5th International Workshop on Peer-to-Peer Systems (IPTPS)*, Santa Barbara, CA, February 2006.

[30] M. Zhong, K. Shen, and J. Seiferas. Non-uniform Random Membership Management in Peer-to-Peer Networks. In *Proc. of the IEEE INFOCOM*, pages 1151–1161, Miami, FL, March 2005.