Random Walks in Peer-to-Peer Networks

Christos Gkantsidis, Milena Mihail, and Amin Saberi
College of Computing
Georgia Institute of Technology
Atlanta, GA
Email: {gantsich, mihail, saberi}@cc.gatech.edu

***Abstract*— We quantify the effectiveness of random walks for searching and construction of unstructured peer-to-peer (P2P) networks. We have identified two cases where the use of random walks for searching achieves better results than flooding: a) when the overlay topology is clustered, and b) when a client re-issues the same query while its horizon does not change much. For construction, we argue that an expander can be maintained dynamically with constant operations per addition. The key technical ingredient of our approach is a deep result of stochastic processes indicating that samples taken from consecutive steps of a random walk can achieve statistical properties similar to independent sampling (if the second eigenvalue of the transition matrix is bounded away from 1, which translates to good expansion of the network; such connectivity is desired, and believed to hold, in every reasonable network and network model). This property has been previously used in complexity theory for construction of pseudorandom number generators. We reveal another facet of this theory and translate savings in random bits to savings in processing overhead.**

Keywords: Peer-to-Peer networks, Statistics, Random Walks, Graph Theory.

I. Introduction

The simulation of a random walk, or more generally a Markov chain, is a fundamental algorithmic paradigm with highly sophisticated and profound impact in algorithms and complexity theory. Furthermore, it has found a wide range of applications in such diverse fields as statistics, physics, artificial intelligence, vision, population dynamics, bioinformatics, among others.

Recently, random walks have been proposed as primary algorithmic ingredients in protocols addressing searching and topology maintenance of unstructured P2P networks. In particular:

(a) Following extensive experimentation, Lv et al. report that searching by simulating random walks is preferable to the standard practice of searching by flooding [1]. They attribute the suitability of random walks to their adaptivity in termination conditions and hence granularity in coverage of the search space (in flooding, increasing the TTL by 1 may increase the space coverage exponentially).

(b) Law and Siu give a distributed algorithm for constructing and maintaining unstructured topologies with very strong connectivity properties, namely constant degree and constant expansion, with $O(\log n)$ overhead per addition of a peer, where n is the number of peers [2]. At a very high level, when a new peer arrives, their protocol would ideally attach the new node to existing peers chosen uniformly at random. They approximate such uniform sampling by simulating $O(\log n)$ steps of a random walk.

What are the analytic reasons of the success of the random walk method? Can we isolate one or two comprehensible analytic primitives that explain the power of the method? Most important, can we translate these primitives to heuristics, or rules of thumb, for the use of the method in P2P network applications?

Independent sampling from the uniform distribution is a primodal statistical, and hence algorithmic primitive. However, it is infeasible to implement in many populations of complex systems, such as the set of nodes of a P2P network. The difficulty arises from the fact that this set is not centrally maintained and it is also quite dynamic. In this paper we make the following argument: *(i) The random walk method is an excellent candidate to simulate sampling for P2P networks, moreover, (ii) the number of simulation steps required can be as low as the number of samples in independent uniform sampling, which translates to constant network overhead, independent of the size of the network.*

In particular, beyond termination adaptivity and space coverage granularity discussed in [1], we believe that the power of the random walk method can be pinned down into two kinds of *analytic* properties:

The *first analytic property*, corresponding to *(i)*, is as follows. Consider a population whose members can be connected by links forming a connected graph. Perform a random walk starting at any state and simulate the random walk for τ steps, Use the state of the random walk at state τ as a sample point. This simulates sampling with arbitrary accuracy, for τ bounded by well defined parameters of the graph. The above is rather intuitive, since we expect that graphs without any "bad" or "bottleneck" cuts will make the walk to "loose memory" and hence reach a random state quite fast. In particular, for a wide range of applications, it is possible to construct sparse graphs so that $\tau = O(\log n)$. Such fast convergence rates translate to practically efficient simulation of uniform sampling.

The *second analytic property*, related to *(ii)*, is substantially more profound and rather counter-intuitive. It states that starting the random walk at a random state, simulating the walk for k steps, and using each visited node as a sample point, we may achieve same statistical properties as k independent uniform samples. The reason why this is counter-intuitive is because of the obvious huge dependencies between successive steps of a random walk.

In this paper we focus on two central issues of P2P networks, namely searching and overlay topology construction. For both problems we have isolated scenaria where independent uniform sampling would have been a good algorithmic primitive. We use k successive states visited during the simulation of a random walk on the P2P network in the place of k independent samples. We compare the performance of sampling by random walk to the performance of the previouly known methods for search and construction. Our results and conclusions are:

(a) For searching relatively popular items, we found, experimentally, that random walk performs better than flooding, for the same number of network messages, in two cases: (i) When peers in the topology form clusters so that the whole topology is arranged in two tiers, the lower representing peer clustering and the higher connecting representatives from each cluster (e.g. super-nodes) to ensure good global connectivity. (ii) When the same search request is re-issued repeatedly, in hopes of finding new peers, while the entire topology has not changed dramatically (less than 40%). We believe that both scenaria are realistic. However, to the best of our knowledge, they have not been considered in previous studies.

We believe that our results in the context of searching are intuitive. Our primary contribution was to formalize set-ups of practical interest and translate our analytic intuition in these set-ups.

(b) For constructing and maintaining a P2P topology with good connectivity properties, we turned to the approach of [2]. As mentioned before, when a new peer arrives, they find a few nearly uniformly random existing peers to connect the new peer by simulating $O(\log n)$ steps of a random walk. This causes $O(\log n)$ network overhead per newly arriving peer. We introduce a daemon construction and connect a newly arriving peer with constant network overhead. Our construction is based on the second analytic property, described earlier in this section. In fact, there are strong dependencies between the edges of peers that arrive closely in time. We give analytic evidence that these local dependencies do not affect the global connectivity properties of the network, up to constant factors. We also give strong experimental evidence that our daemon construction simulates the algorithm of [2] (and other related constructions) (i) with overhead a very small constant per arriving peer, (ii) for networks up to 5M nodes (which is the current believed size of Kazaa; larger experiments were stressing the memory limits of our machines), and (iii) with truly negligible penalty in the quality of the connectivity of the overall topology.

We believe that our results in the context of P2P network construction are particularly surprising. From a practical point of view, they indicate that minimal amount of correctly used randomization suffices to keep a dynamic network well connected. From a theoretical point of view, we believe that our results lead to an exciting new problem and paradigm in the study of the power and necessity of randomness. All of our algorithms in the context of construction were inspired by the second analytic property of random walks.

The isolation of the second property has been one of the most celebrated results in complexity theory [3]–[6]. In complexity theory this property has been used as follows. Consider a randomized algorithm that uses n random bits and has probability of success 1/2. By simulating the algorithm k times we may decrease the failure probability to $1/2^k$. This needs kn random bits. Now think of the nodes of a graph labeled by all 2^n n-bit strings that can be used to simulate a randomized algorithm. Consider a constant degree "expander" graph $\mathcal{H}$ imposed on these 2^n nodes (there are known deterministic constructions of such expander graphs [7] [7]. Start from a uniformly random point of $\mathcal{H}$ and simulate a random walk on $\mathcal{H}$ for k steps. This requires $O(n)$ random bits to pick the initial point, and $O(k)$ bits to perform the walk. Then simulate the randomized algorithm on the k n-bit strings visited by the random walk. The failure probability is $O(1/2^k)$, and yet we used only $O(n + k)$ random bits to perform the experiment! This idea is the basis of several pseudorandom number generators with provably good performance. In some sense, our work can be viewed as translating a complexity result, mostly known in the context of savings in random bits, into savings in overhead and improved performance in a practical networking context.

The balance of the paper is as follows: In Section II, we give the supporting theory, comparing coupon collection and Chernoff bounds to the corresponding statements for random walks. All the technical parts of this section are known; their synthesis and relevance in the context of networking is new. In Section III, we use random walks to perform searching in P2P networks and compare this approach to searching using flooding and uniform sampling. Note that flooding corresponds to breath first search, whose statistics are not as well quantified as those of random walks. In Section IV, we describe two algorithms for distributed construction of P2P topologies with good expansion properties. We stress that our experiments in Sections III and IV are on the size of current P2P networks. Many implementational and other details are suppressed to emphasize clearly the new ideas and for lack of space. We conclude in Section V.

II. Statistical Estimation and Random Walks

In this section we focus on the statistical properties of sampling performed (a) by ideal independent draws, and (b) by simulating a random walk. In the case of independent sampling we are interested in the number of samples sufficient to achieve a certain statistical property. In the case of random walks we are interested in the number of simulation steps sufficient to achieve the same statistical property.

For comparison, we consider two common abstractions, namely the coupon collection problem and Chernoff bounds for independent Bernoulli trials; these abstractions refer to sampling by independent draws. For sampling by random walk simulation, we consider the cover time which is the suitable analogy to coupon collection, and the trajectory sample average which is the suitable analogy to independent Bernoulli trials.

We observe that, for both abstractions, the overhead of the random walk simulation method is determined only by the second eigenvalue of the probability transition matrix of the random walk. In particular, when the second eigenvalue is constant (independent of the size of the graph) the random walk method achieves the same statistical characteristics as independent sampling, up to $O()$ notation.

The reason why the second eigenvalue provides such clean characterizations is that it is intimately related to global connectivity properties of the graph, namely expansion and conductance. Intuitively, expansion and conductance express the worst-case cuts of the graph, and it is natural to expect that when the graph does not have bad cuts a random walks approaches its stationary distribution very fast, and hence sampling by random walk mimics independent sampling well.

A. Coupon Collection and Chernoff Bounds

The *coupon collection problem* is the following: Suppose that there are n distinct types of coupons. At each step, we draw a coupon whose type is uniformly distributed among all n types. Let T_n be the time by which we have encountered coupons belonging to all n distinct types. It is well known [8] that

$$E[T_n] = 1 + \frac{n}{n-1} + \frac{n}{n-2} + \ldots + n = O(n \log n) \quad . \quad (1)$$

Let δ be a constant, $0 < \delta < 1$. Let $T_{\delta n}$ be the time by which we have encountered coupons belonging to δn distinct types. It is also well known that

$$E[T_{\delta n}] = 1 + \frac{n}{n-1} + \ldots + \frac{n}{n-\delta n+1} = \frac{1}{1-\delta} O(n) \quad . \quad (2)$$

We proceed with an outline of *Chernoff bounds* [9]. Let $X_1, \ldots, X_k$ be independent Bernoulli trials with $\Pr[X_i = 1] = p$ and $\Pr[X_i = 0] = 1\text{–}p$, $0 \le p \le 1$, $1 \le i \le k$. Let $X = \sum_{i=1}^{k} X_i$, hence $E[X] = kp$. In a searching context, where p denotes the probability that a randomly drawn object has a desired property, we are interested in the probability that the property is found in substantially fewer draws than its frequency in the search space. This corresponds to the event $X \le (1-\epsilon)kp$, for $0 < \epsilon < 1$. For this event, Chernoff bounds are

$$\Pr[X \le (1-\epsilon)kp] \le e^{-\frac{\epsilon kp}{2}} \quad . \quad (3)$$

In a measurement context, where p denotes the fraction of objects satisfying a certain property, we are interested in the quality of the estimator X/k for p. Now, for $0 < \epsilon < 0.9$, Chernoff bounds are:

$$\Pr\left[|\frac{X}{k} - p| \ge \epsilon p \right] \le 2e^{-\frac{\epsilon^2 kp}{20}} \quad . \quad (4)$$

B. Random Walks, Convergence, Cover Time and Trajectory Sample Average

Let $G(V, E)$ be an undirected connected graph, $|V| = n$. Let d_i denote the degree of vertex i, $1 \le i \le n$. Let $d_{\min} = \min_{1 \le i \le n}\{d_i\}$. Let $A = \{a_{ij}\}$, $1 \le i, j \le n$, be the adjacency matrix of G. Let P be the transition matrix of the random walk on G, where a particle that is on vertex i at time t, moves to a neighbor of i at time $t+1$, chosen uniformly at random among all neighbors of i. It is well known and easy to verify that the above random walk has a unique stationary distribution $\vec{\pi}$, in the sense that $\vec{\pi}P = \vec{\pi}$, with $\pi_i = d_i/2|E|$, $1 \le i \le n$, and let $\pi_{\min} = d_{\min}/2|E|$.

Let f be a 0-1 function on V, $f : V \xrightarrow{f} \{0, 1\}$. Let p be the probability mass of vertices that take the value 1 under f under the stationary distribution $\vec{\pi}$, which is the same as the mean of f under $\vec{\pi}$:

$$p = \sum_{v \in V : f(v) = 1} \pi_v = \sum_{v \in V} f(v)\pi_v \quad . \quad (5)$$

Of particular interest are the following three metrics: (a) *Convergence rate*, which is the rate with which the random walk approaches the stationary distribution. (b) *Cover time*, which is the time when the random walk has visited all vertices at least once. This is analogous to the coupon collection abstraction, and we wish to have bounds comparable to (1) and (2). (c) *Trajectory sample average*, which is the rate with which the value of f, averaged over successive vertices of a trajectory of the random walk, approaches p. This is analogous to the Chernoff bound abstraction, and we wish to have bounds comparable to (4). In the next paragraph we point out bounds for all the above metrics in terms of the second eigenvalue of P.

C. Bounds in terms of the Second Eigenvalue

In general, a vector $\vec{x}$ is an eigenvector of P with eigenvalue λ if and only if $\vec{x}P = \lambda\vec{x}$. Thus, $\vec{\pi}$ is an eigenvector of P with eigenvalue 1. It is well known that P has n real eigenvectors with corresponding eigenvalues $1 = \lambda_1 > \lambda_2 \ge \ldots \ge \lambda_n \ge -1$ [10] [11]; the strict separation of the first and second eigenvalues follows from the connectivity of G. The first eigenvalue $\lambda_1 = 1$ which corresponds to eigenvector $\vec{\pi}$ characterizes stationarity. We may also assume that $|\lambda_2| > |\lambda_n|$ (large negative eigenvalues concern strong periodicities, like bipartiteness, which we may exclude for the purposes of this paper).

Consider a random walk on G according to the transition matrix P, starting from an arbitrary vertex, or an arbitrary distribution on V. Let y_t be the vertex that the random walk visits at time t, $1 \le t \le \infty$. To bound the convergence rate of the random walk we focus on the so-called *variation distance* which, at time t, is $\Delta(t) = \max_{S \subset V} |\Pr[y_t \in S] - \vec{\pi}(S)|$. The following is known [12]:

$$\Delta(t) \le \pi_{\min}^{-1} \lambda_2^t.$$

Let C_n be the time by which the above random walk visits all the vertices of G. [13] [14] show:

$$\begin{aligned} E[C_n] &\le O\left(\pi_{\min}^{-1} \log n/(1-\lambda_2)\right) \\ &= O\left(n \log n/(1-\lambda_2)\right) \quad , \quad \text{for} \quad \pi_{\min} = \Omega(\tfrac{1}{n}). \end{aligned} \quad (6)$$

Compare (6) to (1) and realize that, for constant λ_2, they both solve coupon collection in the same order of magnitude.

Let $C_{\delta n}$ be the time by which the random walk visits δn distinct vertices of G, for some constant δ, $0 < \alpha < 1$. It is

straightforward to derive from [13] [14] (and is folklore among probabilists) that

$$\begin{aligned} E[C_{\delta n}] &\leq \frac{1}{1-\delta} O\left(\pi_{\min}^{-1}/(1-\lambda_2)\right) \\ &= \frac{1}{1-\delta} O\left(n/(1-\lambda_2)\right) \quad , \quad \text{for} \quad \pi_{\min} = \Omega(\tfrac{1}{n}). \end{aligned} \tag{7}$$

Compare (7) to (2) and realize that, for constant λ_2, they both solve partial coupon collection in the same order of magnitude.

Recall that y_t denotes the vertex that the random walk visits at time t. Let $Y_t = f(y_t)$. Suppose that we simulate the random walk for

$$\tau = \frac{\log \pi_{\min}^{-1}}{1-\lambda_2} \geq \frac{\log \pi_{\min}^{-1}}{\log \lambda_2^{-1}} \tag{8}$$

steps. Very roughly, this guarantees good approach to stationarity according to the bound on Δt. Then use the next k steps as sample points. We thus let $Y = \sum_{t=\tau+1}^{\tau+k} Y_t$. The particularly strong result is that, using Y/k as an estimator for p, is of the same quality as Chernoff bounds. Thus, despite the local dependencies introduced by consecutive steps of the random walk, the overall distribution of the vertices visited by the random walk is well spread across the sample space. In particular, (9) below is to be compared to (4):

$$\Pr[|\frac{Y}{k} - p| \geq \epsilon p] \leq 8e^{-\frac{\epsilon^2 k p^2 (1-\lambda_2)}{20}} \quad . \tag{9}$$

The above result was obtained (in increasingly stronger forms and referring to pseudorandom number simulation) in a sequence of celebrated complexity theory papers [3]–[6]. The version that we give above is from [6].

D. Second Eigenvalue, Expansion and Conductance

For $S \subseteq V$, define the cutset of S, $C(S)$, as the set of edges with one endpoint in S and the other endpoint is $\bar{S}$. Define the *volume* of S as the sum of the degrees of vertices in S: $\mathrm{vol}(S) = \sum_{v \in S} d_v$. The *expansion*, ϕ, and the *conductance*, Φ, of G are:

$$\phi = \min_{\substack{S \subset V \\ |S| \leq |V|/2}} \frac{|C(S)|}{|S|} \quad , \quad \Phi = \min_{\substack{S \subset V \\ \mathrm{vol}(S) \leq \\ \mathrm{vol}(V)/2}} \frac{|C(S)|}{\mathrm{vol}(S)}. \tag{10}$$

In addition, the following bound is known [12]:

$$1 - 2\Phi \leq \lambda_2 \leq 1 - \frac{\Phi^2}{2} \quad . \tag{11}$$

Finally, realize that in graphs where we have bounds on the minimum and maximum degrees, both expansion, conductance and eigenvalues are easily related. In particular, in a family of regular graphs, if any of these metrics is a constant then all of them are constants.

In summary, for families of graphs where λ_2 is constant, consecutive states of random walks are excellent candidates to approximate independent uniform sampling. Since, λ_2 constant is equivalent to expansion ϕ constant, and constant expansion is equivalent to good global connectivity, it is reasonable to try the random walk approach in communication networks. Good global connectivity is desired and believed to hold in all reasonable networks and network models [15]–[20].

III. Searching

In this section we study the performance of searching using flooding and random walks, and compare the two methods to each other and to a baseline case of independent uniform sampling. We measure the performance in terms of the average number of distinct copies of an item located in the search, the probability of not finding any copy of the item, and the number of messages that the searching algorithm uses. We show experimentally that searching by random walks is better than flooding, if at least one of the following conditions holds:

- The user issues multiple search requests for the same item and between two consecutive requests the topology changes relatively slowly (in the sense that two consecutive snapshots of the topology are highly correlated).
- There is peer clustering. That is, there are communities in the topology, with dense connectivity between peers in the same community and sparse connectivity between peers of different communities.

We believe that the two scenarios introduced above are important for the following reasons:

In practice, when a user issues a request, the user (or, the system on behalf of the user) re-issues the same request multiple times hoping to locate more sources. Consecutive floodings take advantage of the changes in the topology so as to discover more sources. If the topology however remains mostly unchanged between consecutive requests, then the new floodings will mostly discover sources already known. On the other hand, for the same number of messages, the random walk follows totally different trajectories and has better chances to discover new sources. Note that in previous work, searching has been always modeled as an one time process. However, we believe that studying searching under multiple requests and a changing topology is realistic and important.

The motivation behind studying peer clustering becomes clear if we consider the process by which the P2P network is formed. Each peer keeps a cache of other peers and picks its neighbors from its cache. The cache is populated by the addresses of peers that answered previous queries [21]. Thus, intuitively, the cache contains addresses of peers that have similar interests. It is therefore reasonable to expect that this process leads to the formation of communities of users. The exact process by which P2P networks are formed is largely unknown and thus peer clustering is at this point only a hypothesis. But, we believe that it is a fair hypothesis based both on our practical experience with P2P systems, and on the observation that most networks grown in a decentralized way exhibit strong clustering properties. See also [22]–[25] and for related discussion [26].

The rest of the section is organized as follows. First, we give the methodology. Then, we discuss the most simple case of a topology without clustering that does not change with time. In this scenario, we find that flooding and random walk behave

similarly. Next, we examine topologies with peer clustering. Subsequently, we examine multiple re-issues of a request in topologies that change with time. In the last part we discuss power-law graphs and real topologies. Random walks behave better than flooding in most cases of interest.

A. Methodology

In all our experiments we assume that copies of the item to be discovered populate $\alpha\%$ of the peers, where α is a parameter with $0.01\% \leq \alpha \leq 0.1\%$; tnis represents items that are not rare. Assuming that a single search reaches 10000 distinct nodes, a typical value of the horizon of a user in the Gnutella network [21], the search will result, in expectation, in 1 to 10 distinct copies found for the range of α's we have experimented with. The performance of each searching technique is measured as the number of distinct copies located when simulating the searching algorithm from *a randomly chosen peer* of the topology. To make statistically robust conclusions, we repeat the experiment from a set of randomly chosen peers, typically 500 peers, and study the distribution of the number of distinct copies located (hits).

a) Performance Metrics: A metric that summarizes the distribution of the hits is the mean. Of equal importance, however, is the discrepancy around the mean, and failure probability (probability of no copies discovered). Even when the means of random walks and flooding are the same, it is almost always the case that the discrepancy and failure probability of random walks are substantially better than flooding (e.g. see Figure 2). We therefore measure *Mean*, standard deviation *Std*, and *Failure probability*.

b) Cost: We measure the cost of each searching technique as the number of messages or queries performed during the search. When comparing different algorithms, it is always under the assumption of using the same number of messages.

c) Peer-to-peer topologies: Available topologies of current P2P networks are limited in size and of questionable quality due to the collection method (topologies from [27] have only 30-40K nodes, when current P2P networks have hundreds of thousands and perhaps millions of users). We have therefore experimented on synthetic topologies of up to 1 million nodes. Experimenting with extremal synthetic topologies has the additional advantage of facilitating the demonstration of general principles.

We have used the following models to generate synthetic P2P topologies:

- Flat regular expanders. This is a canonical example of regular graphs with good expansion properties. We use expanders since expansion is desired and believed to hold in every reasonable network and network model [2], [18], [19]. We have used 6-regular expanders.
- Two-tier topologies with clustering. To study the effects of peer clustering we have started by constructing a number of isolated regular expanders that correspond to the clusters. Then, from each cluster we pick a small number of nodes at random and connect them using another regular expander.
- Power-law graphs. Many important networks that arise in a decentralized fashion are known to have power-laws [16], [24], [28]. Some researches argue that P2P topologies may also possess heavy tails [29]. We have used the standard model of growth with preferential connectivity to generate power-law random graphs (this model runs in linear time and hence can efficiently generate graphs of very large size).
- Samples of real topologies. We have used partial views of the Gnutella topology made available in [27]. These topologies are limited in the number of peers (around 35K) and of questionable accuracy, since the topology evolves during the topology discovery process, some peers are uncooperative and for other practical reasons. Because of their very limited size, our results are inconclusive.

d) Dynamic Topologies: The dynamic nature of P2P topologies is a crucial parameter of these systems [30]. However, very few things are known about the way these topologies evolve over time. To model the dynamic nature of the P2P topologies we have used the following heuristic: We perform a number of "rewirings"; for each rewiring we pick two edges uniformly at random and exchange their end points. The number of rewirings is a parameter that is related to the speed by which the topology is changing. In our experiments, the measurements are happening before and after the rewirings and not during the process of changing the topology. The size of the topology remains unchanged during the rewirings, because we want to capture only the effects of changes in the connectivity and not in the number of peers.

In our experiments we measure the speed by which the topology is changing as the ratio of the number of links changed by performing rewirings over the total number of links. We have experimented with ratios in the range from 2% to 40%. The rate of change in current P2P networks is not known and is difficult to estimate. However, from our practical experience, we observe that consecutive searches that happen every 10-20min do not result in differences in hosts discovered that would have been expected if a large fraction of the network has changed.

e) Content placement: The straightforward approach is to pick the nodes that will host the copies uniformly at random from the entire population. This is what we have used in our experiments. We have also experimented with cases where the nodes that host the item are close to each other in the topology (content clustering). In our experiments, we have observed that content clustering affects the performance of searching by flooding or random walk much less than peer clustering, or re-issuing of the same request. We therefore do not present this case.

B. Flat Topologies with Uniformly Distributed Content

We start by examining a scenario in which the performance of flooding and random walks is similar. See Table I. We study the performance of issuing a request only once in a flat regular topology of 500K peers. After simulating the

TABLE I

PERFORMANCE OF SEARCHING IN A STATIC TOPOLOGY WITHOUT PEER CLUSTERING.

Attribute	Flooding	RW	Uniform
Mean	8.712	8.796	10.990
Std	3.01	2.93	3.22
Min	1	2	3
Messages	22331	22331	22331
Unique peers	17235	17431	21839

Note: 500K peers, $\alpha = 0.05\%$. Min is the minimum number of hits over all searching requests. Unique peers is the number of distinct peers discovered during the search. Observe that flooding and RW have very similar performance, while uniform sampling is better.

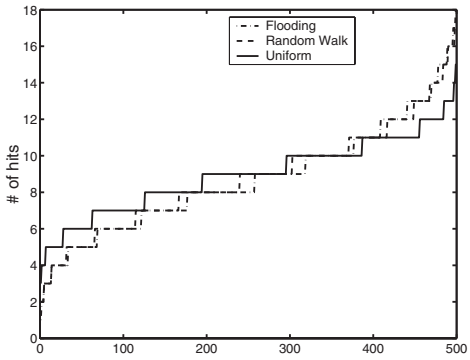


Fig. 1. Sorted number of hits when searching from 500 randomly chosen peers. Topology of 500K peers, $\alpha = 0.05\%$. Observe that flooding and random walk have very similar performance.

flooding algorithm with a time to live (TTL) of 5 and counting the number of messages, we run the random walk algorithm and configured it to use the same number of messages. Observe that the mean and minimum numbers of hits, and the standard deviation of the hits distribution of both flooding and random walk are roughly the same, while independent uniform sampling is better. Moreover the entire distribution of hits, given in Figure 1, is similar for both random walk and flooding.

We have experimented with topologies of various sizes and for various popularities of files, with $0.01\% \leq \alpha \leq 0.1\%$, and found that always the performance of flooding and random walk are similar, when both are allowed to use the same number of messages.

Observe that compared to the study of [1] we use only one walker. The results were similar when using a larger number of walkers assuming that the total number of messages stays the same. The use of more walkers decreases the user-perceived delay, which is a parameter not studied in this paper.

C. Topologies with Peer Clustering

We now examine a topology with well separated communities of peers and show that the random walk method has better performance than flooding. The example topology is constructed as follows. We generate five flat regular graphs each with size 40K. From each topology we pick 1000 nodes at random (for a total of 5K nodes) and construct another flat

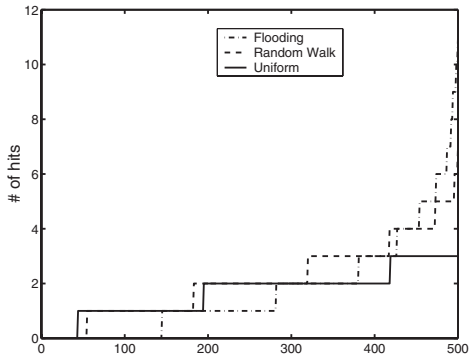


Fig. 2. Sorted number of hits in a topology with peer clustering. The distribution of random walk is more concentrated around the mean. Topology of 200K peers, $\alpha = 0.05\%$.

regular graph on the selected nodes. The final P2P network is the union of all topologies.

The performance of each searching algorithm for different content popularities is given in Table II. First, observe that the average number of hits using flooding is slightly worse than using random walks but, given the large standard deviations, it could be argued that the differences are not statistically significant. Even though the two schemes behave similarly with respect to the average number of hits, they have totally different behavior when it comes to the failure rate and the minimum number of copies found. For $\alpha = 0.01\%$, flooding failed in 28.8% of the cases and random walk in only 10.8%, an improvement of nearly a factor of 2.

The fundamental difference between the two searching algorithms is that the number of hits in the case of random walks appears more concentrated around the mean (observe also the entire distribution of hits in Figure 2). Indeed, the standard deviation in the case of random walks is much smaller compared to the standard deviation of flooding. Obviously, the optimal concentration around the mean is achieved by uniform sampling. The basic strength of random walks, following Section II, is precisely that they resemble uniform sampling in a quantifiable way.

D. Re-issuing the Same Query

In this section, we study the performance of flooding and random walks in flat regular topologies under the assumption that users issue the same query multiple times while the topology is changing. Realize that dynamic topologies favor only flooding, while random walks will be largely unaffected. In fact, in the worst case where the topology is not changing, re-issuing a query k times, by flooding, does not find new copies. On the other hand, according to (7) and (9), increasing the length of the random walk by a factor of k has substantial impact.

The performance of the different algorithms for topologies of various sizes and for various values of α is given in Table III. Our experimental methodology is as follows: Each peer initiates a searching request and waits for the results. Then, we

TABLE II

PERFORMANCE OF SEARCHING IN A TOPOLOGY WITH PEER CLUSTERING.

Method	$\alpha = 0.01\%$				$\alpha = 0.05\%$				$\alpha = 0.10\%$			
	Mean	Std	Min	Failure	Mean	Std	Min	Failure	Mean	Std	Min	Failure
Flooding	1.754	1.91	0	28.8%	9.308	6.44	1	0.0%	18.192	11.04	5	0.0%
RW	2.124	1.38	0	10.8%	10.860	3.21	2	0.0%	21.764	4.54	10	0.0%
Uniform	2.676	1.52	0	6.6%	13.496	3.26	5	0.0%	27.274	4.87	15	0.0%

Note: Topology of five clusters for a total of 200K peers. $\lambda_2 = 0.9956$.

TABLE IV

PERFORMANCE OF SEARCHING IN DYNAMIC TOPOLOGIES AS A FUNCTION OF THE RATE OF CHANGES.

Links Changed	Flooding			Random Walk		
	Mean	Std	Failure	Mean	Std	Failure
2%	0.488	0.67	60.6%	1.398	1.14	24.6%
4%	0.644	0.82	53.0%	1.382	1.11	23.6%
10%	0.888	0.86	38.0%	1.450	1.11	21.4%
20%	1.162	0.99	27.6%	1.456	1.12	20.8%
40%	1.460	1.12	20.0%	1.378	1.13	23.8%

change the topology by performing rewiring operations to 2% of the links. Then, each peer initiates a new searching request. We repeat the process four times, simulating consecutive queries. In the end, we count the number of distinct items found for each peer.

Table III indicates that random walks have better performance compared to flooding with respect to both the average number of hits and the probability of failure. The average number of hits for random walks was at least three times better compared to the same number for flooding. Also, the failure probability dropped substantially. This great performance improvement was expected since, even though we change a certain number of links, the overall topology remains relatively stable and successive flooding searches do not result in many new items found. On the other hand, prolonging the random walk, or, successive random walks from the same peer, follow totally different sampling paths and have better chances of locating new copies of the requested item.

The performance of successive searches depends on the number of topology changes that take place between the consecutive searches. We have studied this effect and report the results in Table IV. We observe that the performance of flooding increases as the rate of topological changes increases. For very fast rates of change (40% at each step in our experiment), the performance of flooding becomes comparable to that of random walks, since effectively the neighborhood of each node changes almost completely between consecutive searches. On the other hand, the performance of random walk remains relatively unaffected by the changes in the topology.

E. Real topologies and topologies with power-law statistics

In the previous analysis we have experimented on flat regular graphs. Similar results hold for topologies with heavy-tailed statistics as well as real topologies. In Figure 3, we show the distribution of hits for a topology of 500K nodes generated with the model of growth with preferential connectivity [31], and for a real Gnutella topology taken from [27]. Again, observe that the distribution of hits in the case of random walk is more concentrated around the mean compared to flooding. Indeed, in the real topology, the mean number of hits, the standard deviation and the failure rate were 0.514, 2.15 and 81% respectively in the case of flooding in the real topology, and 0.538, 0.73 and 59% in the case of random walk. Similar results apply for the graph grown with preferential connectivity.

Observe that the very small TTL used for flooding in the case of the real topology was due to the small size of the topology. Increasing the TTL to 3, would have resulted in reaching almost half of the nodes with flooding and would have skewed the statistics. It is more realistic to expect that searching visits only a small portion of the graph.

In conclusion, we expect that our results apply to all graphs with good expansion property as expected from the theoretical section. In addition, we expect that typical networks, like P2P networks, have good expansion properties; otherwise, they would not have scaled easily from a few tens of thousands to a few million nodes.

IV. CONSTRUCTION

In this section we turn our attention to construction and maintenance of well connected P2P topologies. Following the spirit of the previous sections, as well as the work of [2], [17], [18], [20], we translate good connectivity to good expansion, conductance, and separation of λ_2 from 1. We further translate the fact that the construction concerns a P2P network to the following conditions:

- **(i)** Peers arrive and leave the network dynamically.
- **(ii)** The algorithm must be *strongly* or *weakly* decentralized. By strong decentralization we mean that there is no central server. In weak decentralization there is a constant number of central servers. However, the computational resources of each central server are of the same order of magnitude as those of an average peer. In other words, a typical peer can simulate the behavior of a central server.
- **(iii)** We wish to achieve low *network overhead*, in terms of messages, per addition or deletion. In the rest of section. we deal only with additions. Deletions can be handled as in [2].

In paragraph IV-A we shall revisit the known randomized algorithms for constructing sparse graphs with good expansion properties, without being concerned about conditions **(i)** and **(ii)**. We call these *baseline constructions*. The schemes for

TABLE III

PERFORMANCE OF SEARCHING IN DYNAMIC TOPOLOGIES.

A. Flooding

Size (K)	$\alpha = 0.01\%$				$\alpha = 0.05\%$				$\alpha = 0.10\%$				λ_2
	Mean	Std	Min	Failure	Mean	Std	Min	Failure	Mean	Std	Min	Failure	
100	0.488	0.67	0	60.6%	2.294	1.45	0	8.6%	4.566	2.08	0	1.4%	0.79
300	0.412	0.64	0	66.2%	2.350	1.57	0	9.4%	4.918	2.26	0	0.4%	0.79
500	0.550	0.75	0	58.6%	2.562	1.68	0	8.8%	4.992	2.27	0	0.6%	0.79
1000	0.494	0.62	0	60.0%	2.684	1.72	0	8.7%	5.032	2.31	0	0.2%	0.80

B. Random Walk

Size (K)	$\alpha = 0.01\%$				$\alpha = 0.05\%$				$\alpha = 0.10\%$				λ_2
	Mean	Std	Min	Failure	Mean	Std	Min	Failure	Mean	Std	Min	Failure	
100	1.398	1.14	0	24.6%	7.058	2.40	1	0.0%	14.076	3.30	5	0.0%	0.79
300	1.436	1.12	0	20.2%	7.396	2.62	1	0.0%	14.894	3.87	5	0.0%	0.79
500	1.562	1.19	0	19.8%	7.634	2.78	1	0.0%	15.152	3.88	7	0.0%	0.79
1000	1.518	1.20	0	22.4%	7.544	2.71	1	0.0%	14.982	3.91	4	0.0%	0.80

C. Uniform Sampling

Size (K)	$\alpha = 0.01\%$				$\alpha = 0.05\%$				$\alpha = 0.10\%$				λ_2
	Mean	Std	Min	Failure	Mean	Std	Min	Failure	Mean	Std	Min	Failure	
100	1.734	1.18	0	13.8%	8.634	2.83	1	0.0%	17.210	3.75	7	0.0%	0.79
300	1.864	1.23	0	12.6%	9.212	2.90	1	0.0%	18.496	4.07	9	0.0%	0.79
500	1.872	1.38	0	15.2%	9.486	2.98	2	0.0%	18.924	4.21	10	0.0%	0.79
1000	1.876	1.34	0	14.2%	9.482	3.14	0	0.2%	18.850	4.34	8	0.0%	0.80

constructing expander graphs under conditions **(i)** and **(ii)** explicitly simulate a baseline construction.

All known baseline algorithms construct an expander essentially by choosing the edges incident to a vertex uniformly at random, and independently for each vertex. This facilitates the probabilistic arguments that are subsequently used to establish expansion. We review the baseline probabilistic argument in Theorem 4.1.

Thus, when constructing an expander on n vertices, a baseline construction uses $O(\log n)$ random bits per edge. In a distributed setting, when vertices arrive dynamically and a new vertex needs to extend an edge to an existing vertex, one may use $O(\log n)$ steps of a random walk on the existing graph to find a random existing vertex, thus simulating the baseline construction with $O(\log n)$ message overhead on the network. This is the approach of [2]. (Pandurangan et al. use the fact that deletions happen in a random way, and they use the "randomness" of deletions to connect new vertices [17]).

In paragraph IV-B, Theorem 4.2, we show, analytically, that a certain baseline construction achieves non-trivial expansion properties using constant number of random bits per new edge. When translated to overhead in network resources, this gives a heuristic to construct an expander with constant message overhead per new vertex. In particular, instead of taking $O(\log n)$ steps of a random walk per newly arriving vertex, we do the following. We keep a constant number of daemons which continuously simulate a random walk on the existing network and use the vertices visited by the daemons every c steps as sample points, where c is a constant. We stress that we do not wait $O(\log n)$ steps until the daemon randomizes. In paragraph IV-C we report that, in experiment, the method achieves constant separation of λ_2 from 1 for c as small as 1 (sampling consecutive vertices visited by the daemons). The eigenvalue gap depends on c and on the average degree of the constructed graph, but does not depend on the size of the constructed graph, for n as large as 5M vertices. Note that the current size of Kazaa is thought to be 2M to 4M. (Performing experiments for more than 5M vertices was stressing the memory limitations of our machines.)

A. Baseline Construction of Expander Graphs

A_{BASE} is the following construction: On input n, the number of vertices, and d, the degree of every vertex, each vertex, independently, picks d vertices independently and uniformly at random among the set of all vertices, and connects with an (undirected) edge to each one of these vertices. Thus the total number of edges is nd and the expected degree of a vertex is $2d$. (It can be easily seen, using (4), that all vertices will have degree at most $O(\log n)$, almost surely.)

When we insist on all vertices having the same degree, then we simulate A_{BASE} by picking random perfect matchings, or random Hamilton cycles. In particular, $A_{\mathcal{M}}$ is the following construction: Pick d perfect matchings on n vertices independently and uniformly at random among all perfect matchings (assume w.l.o.g. that n is even), and consider the union of these perfect matchings. Finally, $A_{\mathcal{H}}$ is the following construction: Pick d Hamilton cycles on n vertices independently and uniformly at random among all Hamilton cycles, and consider the union of these Hamilton cycles.

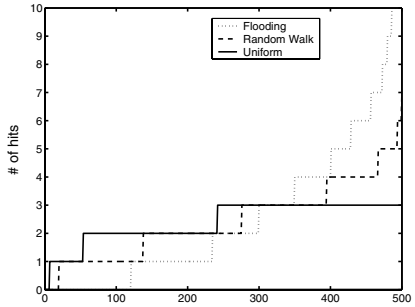
Theorem 4.1 (Folklore.): Let $G(V, E)$ be the graph constructed by A_{BASE}. $G(V, E)$ is an expander, with high probability. In particular, there is a positive constant $\alpha < 1$ such that

$$\Pr[\min_{S \subset V, |S| \leq |V|/2} \frac{|C(S)|}{|S|} \geq \alpha] \geq 1 - o(1) \quad . \qquad (12)$$

Proof: For a positive constant α, we say that a set of vertices S with $|S| \leq |V|/2$ is Bad if and only if $|C(S)| \leq \alpha|S|$. We will show that there exists a positive constant α such that

$$\Pr[\exists \ \mathsf{Bad} S] \ \leq \ o(1) \ . \qquad (13)$$

A. Growth with preferential connectivity model with 500K nodes. (TTL=4)



B. Sample from the Gnutella network with 36K nodes. (TTL=2)

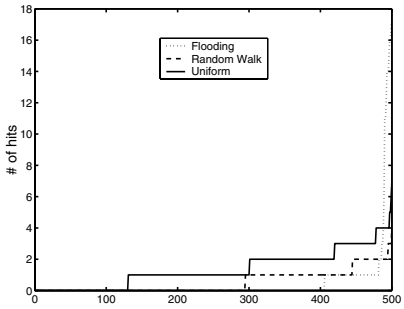


Fig. 3. Performance of searching in A) a network with heavy-tailed statistics, and B) in a real topology. The small TTL in the real topology is due to the fact that using a larter TTL would have resulted in reaching with flooding almost half of the nodes, which is unrealistic.

The left hand side of (13) is

$$\sum_{k=1}^{n/2} \Pr[\exists\ \mathsf{Bad}S,\ |S|=k]\ . \tag{14}$$

Let us fix k in the above range. There are at most $\binom{n}{k}$ sets of vertices of cardinality k. We hence need to bound

$$\sum_{k=1}^{n/2} \binom{n}{k} \Pr[\text{a fixed set } S, |S|=k, \text{ is } \mathsf{Bad}]\ . \tag{15}$$

We may now assume that the set S is fixed. Let $B \subset S$ be the set of vertices in S that chose to connect to vertices in $\bar{S}$. In order for S to be Bad, the cardinality $|B|$ must be at most αk. For each cardinality in the range 0 to αk, there are at most $\binom{k}{\alpha k}$ possibilities for $|S|$, and at most αk possibilities for the cardinality of B. We may now assume that the set B is also fixed. Finally, for fixed S and B, the probability that an edge picked by a vertex in $S \setminus B$ connects to a vertex in S is at most k/n, and there are $d(k-\alpha k)$ such edges. We may now write

$$\Pr[\text{fixed } S, |S|=k, \text{ is } \mathsf{Bad}] \le \alpha k \binom{k}{\alpha k} \left(\frac{k}{n}\right)^{dk(1-\alpha)}\ . \tag{16}$$

Combining (14), (15) and (16), (13) gives:

$$\begin{array}{rl} & \Pr[\exists\ \mathsf{Bad}S] \\ \le & \sum_{k=1}^{n/2} \Pr[\exists\ \mathsf{Bad}S,\ |S|=k] \\ \le & \sum_{k=1}^{n/2} \binom{n}{k} \alpha k \binom{k}{\alpha k} \left(\frac{k}{n}\right)^{dk(1-\alpha)} \\ \text{using} & \left(\frac{n}{k}\right)^k \le \binom{n}{k} \le \left(\frac{en}{k}\right)^k \\ \le & \sum_{k=1}^{n/2} \left(\frac{en}{k}\right)^k \alpha k \left(\frac{ek}{\alpha k}\right)^{\alpha k} \left(\frac{k}{n}\right)^{dk(1-\alpha)} \\ \le & \sum_{k=1}^{n/2} e'^k \left(\frac{k}{n}\right)^{k(d(1-\alpha)-1)} \end{array} \tag{17}$$

where $e' = e'(\alpha)$ is a constant. Now the last line of (17) is bounded by $o(1)$ if every term is bounded by $o(n^{-1})$, which is true for any $d \ge 2(1-\alpha)^{-1}$, since $\alpha < 1$. ■

B. Baseline Construction of Expanders with Constant Overhead in Random Bits

Procedure A_{BASE} assumes that when a vertex chooses d other vertices to attach, these choices are independent and uniformly distributed in the integers 1 to n. Now consider the following pseudorandom number generator: A constant degree expander graph $\mathcal{H}$, with second eigenvalue μ_2, is imposed over a set of n points labeled with the numbers 1 to n. We start a random walk on $\mathcal{H}$ from a point chosen uniformly at random, and, whenever the algorithm A_{BASE} needs a random point, we feed it with the current point of the random walk on $\mathcal{H}$. We call this algorithm A'_{BASE}. Realize that the random choices of A'_{BASE} are highly correlated, since they resulted from consecutive vertices visited by the random walk on $\mathcal{H}$. Nevertheless, we are able to establish a non-trivial expansion property for A_{BASE}, which essentially describes constant expansion of relatively large subsets of vertices. Our proof follows by the same probabilistic argument as Theorem 4.1, except we use (9) to bound the probability of correlated bad events:

Theorem 4.2: Let $G(V,E)$ be a graph constructed by A'_{BASE}. There are positive constants α and β, $0 < \beta < .5$, such that any subset S of at least $\beta|V|$ and at most $|V|/2$ vertices has cutset expansion α, almost surely. In particular,

$$\Pr[\min_{S\subset V, \beta|V|\le|S|\le n/2} \frac{|C(S)|}{|S|} \ge \alpha] \ge 1 - o(1)\ . \tag{18}$$

Proof: The reasoning is identical to the proof of Theorem 4.1, up to (16). At this point we use (9): What is the probability that $dk(1-\alpha)$ subsequent points of the random walk on $\mathcal{H}$ corresponded to ending up inside S, while the probability of falling outside S is at least $(n-k)/n \ge 1/2$? We apply (9) with $\epsilon = 1$ and $p = 1/2$ and get

$$\Pr[\text{fixed } S, |S|=k, \text{ is } \mathsf{Bad}] \le \alpha k \binom{k}{\alpha k} 8e^{-\frac{dk(1-\alpha)}{80}(1-\mu_2)}\ . \tag{19}$$

Now the final calculations become

$$\begin{array}{rl} & \Pr[\exists\ \mathsf{Bad}S] \\ \le & \sum_{k=\beta n}^{n/2} \binom{n}{k} \alpha k \binom{k}{\alpha k} 8e^{-\frac{dk(1-\alpha)}{80}(1-\mu_2)} \\ \le & \sum_{k=\beta n}^{n/2} \left(\frac{en}{k}\right)^k \alpha k \left(\frac{ek}{\alpha k}\right)^{\alpha k} 8e^{-\frac{dk(1-\alpha)}{80}(1-\mu_2)} \\ \le & \sum_{k=\beta n}^{n/2} e^{-\prime k} e^{\frac{dk(1-\alpha)}{80}(1-\mu_2)} \end{array} \tag{20}$$

where $e' = e'(\alpha,\beta)$ is a constant, and the last line of (20) is bounded by $o(1)$ for some constant $d = d(\alpha,\beta)$. Note that $k \ge \beta n$ is crucial to bound $(n/k)^k$ by e^k. ■

TABLE V

λ_2 OF $A'_{\mathcal{H}}$ AS A FUNCTION OF SIZE AND NUMBER OF RANDOM WALK STEPS.

Size (K)	Random	$c = \log_2 n$	c=5	c=3	c=1	c=0
10	0.7455	0.7445	0.7452	0.7506	0.7906	0.9999
50	0.7453	0.7457	0.7459	0.7508	0.7924	0.9999
100	0.7452	0.7453	0.7460	0.7504	0.7938	0.9999
500	0.7453	0.7453	0.7463	0.7504	0.7944	0.9999
1000	0.7453	0.7452	0.7462	0.7503	0.7956	0.9999
5000	0.7453	0.7454	0.7462	0.7504	0.8023	0.9999

C. Distributed Construction of Expanders with Constant Overhead on Network Resources

In this paragraph, we study how the concept of Paragraph IV-B can be used to speed up the approach of [2]. We examine two algorithms.

1) $A'_{\mathcal{H}}$: This is an extension of the scheme proposed in [2]. The authors propose a scheme to implement the $A_{\mathcal{H}}$ construction in a distributed, decentralized environment. To ensure random placement of each arriving node in each of the d cycles, they propose techniques to estimate the size of the network n and then perform d random walks of length $O(\log n)$, thus their overhead is $O(\log n)$.

Instead, we keep d daemons, one for each Hamilton cycle. These daemons move freely in the topology. When a new node arrives, it contacts the daemon associated with the i-th Hamilton cycle, for $1 \leq i \leq d$, and inserts itself between the peer that currently hosts daemon i and one of its two neighbors in cycle i. We require that the daemons perform c number of steps before allowing a new peer to join the topology. Observe that in [2] c is $O(\log n)$. In fact, when c is $O(\log n)$, the daemons can be allowed to start from any node of the topology, and this makes the algorithm of [2] fully decentralized.

We measure the quality of the constructed topology by the second eigenvalue of the corresponding transition matrix. See Table V and Figure 4. It is obvious that λ_2 remains constant as the topology scaled from 10K to 5M nodes. On the other hand, λ_2 depended on c. However, notice that for $c = 1$, λ_2 is larger only by 0.05 compared to $c = \log n$. (As a sanity test, when c is 0, that is without randomization, there is no expansion.) λ_2 also depends on d. In Table V we give the case of $d = 3$ corresponding to degree 6. The trends are identical for larger values of d.

2) $A'_{\mathcal{M}}$: The existence of Hamilton cycles in $A_{\mathcal{H}}$ and $A'_{\mathcal{H}}$ is a good property since it guarantees connectivity. Maintaining Hamilton cycles however is difficult. In this paragraph, we consider a distributed implementation of the $A_{\mathcal{M}}$ algorithm which does not require the existence of a special structure in the topology and thus is easier to implement. On the other hand, the price paid is an increased second eigenvalue. Also, the second eigenvalue does not remain relatively constant with the size of the network, but it increases slightly as we increase the number of peers.

Our algorithm works as follows. Again, we maintain d daemons. We model the arrival of a new node, as the arrival of two nodes X and Y, each with degree d. Upon the arrival, X and Y contact the central server to discover the location of the d daemons. Assume that daemon i is located at node A, and that the i-th neighbor of A is B. The connection between A and B is teared down and the new i-th neighbor of A is X and the new i-th neighbor of B is Y. Between each arrival, the daemons move c steps.

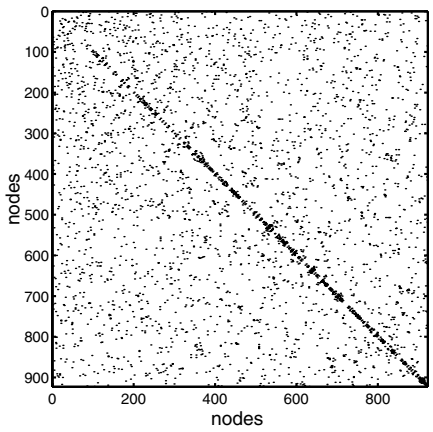


Fig. 4. The connectivity matrix of a topology constructed using $A'_{\mathcal{H}}$ for $c = 1$. The strong dependencies are reflected in the concentration along the diagonal. However, there are many points away from the diagonal and the picture appears random.

TABLE VI

λ_2 OF $A'_{\mathcal{M}}$ AS A FUNCTION OF SIZE, DEGREE d AND NUMBER OF RANDOM WALK STEPS c.

Size (K)	d=4 c=1	d=4 c=2	d=4 c=5	d=4 c=10	d=6 c=1	d=6 c=5	d=6 c=10
1	0.9754	0.8982	0.8711	0.8600	0.7782	0.7385	0.7392
10	0.9893	0.9131	0.8732	0.8654	0.7854	0.7468	0.7443
50	0.9939	0.9144	0.8777	0.8670	0.8015	0.7471	0.7450
100	0.9929	0.9312	0.8925	0.8673	0.8273	0.7470	0.7456
500	0.9969	0.9482	0.8833	0.8679	0.8332	0.7472	0.7454
1000	0.9995	0.9421	0.8861	0.8679	0.8287	0.7476	0.7455
5000	0.9996	0.9504	0.8846	0.8677	0.8348	0.7473	0.7454

The performance of the topology constructed by our algorithm, measured in terms of λ_2, for various sizes of the topology, values of d, and c are given in Table VI. Observe that the second eigenvalue in this construction is a function of the degree, the number of steps and the size of the topology. However, for degree 6 and for $c = 1$, the second eigenvalue for $5M$ nodes is 0.8348 which is comparable to the corresponding entrance value in Table V, which is 0.8023. The interpretation is that for current sizes of the network and for degree at least 6, both methods achieve equally good results.

V. SUMMARY

In this paper we focus on the power and efficiency of sampling using random walks in peer-to-peer networks. We provide theoretical justification in support of random walks as a primitive operation for P2P networks.

We argue that, in the context of searching, random walks are superior to flooding in two cases of practical interest. Related work was previously done in [1].

For construction of dynamic P2P topologies, we use random walks to add new peers with constant overhead. This improves the algorithm of [2]. The construction of [2] is strongly decentralized; however, our construction is weakly decentralized. It is an open question how to design a strongly decentralized construction with constant overhead. Both of these algorithms handle deletions much less effectively than additions. Handling deletions efficiently is an interesting open question.

In practice the performance of searching and constructing methods depends on parameters not studied in this work; some of them are discussed in [30]. It is interesting further work to study the effect of these parameters on random walks. We have also suppressed many implementational details. In practice, we expect adaptations of the random walk methods and in general a hybrid between random walks and other methods.

Theorem 4.2 shows constant expansion of large sets. We can also show expansion $\Omega(1/\log n)$ for all sets, large and small. We believe that constant expansion holds for all sets. Obtaining a proof is an open problem.

Theorems 4.1 and 4.2 concern the so-called base-line constructions, where the random choices are obtained by a random or a pseudo-random number generator. Of particular interest are the real constructions $A'_{\mathcal{H}}$ and $A'_{\mathcal{M}}$ of Section IV-C, where the random choices are generated by the growing network itself. It would be very interesting to prove expansion for these constructions.

REFERENCES

[1] 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 (ICS'02)*. ACM, 2002.

[2] Ching Law and Kai-Yeung Siu, "Distributed construction of random expander networks," in *Proc. Infocom*. IEEE, 2003.

[3] M. Ajtai, J. Komlos, and E. Szemeredi, "Deterministic simulation in logspace," *Proc. 19th ACM Symp. on Theory of Computing (STOC)*, 1987.

[4] D. Aldous, "On the markov chain simulation method for uniform combinatorial distributions and simulated annealing," *Probab. Engrg. Inform. Sci., 1, pp 33-46*, 1987.

[5] R. Impagliazzo and D. Zuckerman, "How to recycle random bits," *Proc. 30th IEEE Symp. on Foundations of Computer Science (FOCS)*, 1989.

[6] D. Gillman, "A chernoff bound for random walks on expander graphs," *Proc. 34th IEEE Symp. on Foundations of Computer Science (FOCS93)*, vol. SIAM J. Comp., Vol 27, No 4, 1998.

[7] O. Gabber and Z. Galil, "Explicit constructions of linear-sized superconcentrators," *Journal of Computer and System Sciences*, vol. 22, pp. 407–420, 1981.

[8] William Feller, *An Introduction to Probability Theory and Its Applications*, vol. I, John Willey & Sons, 3rd edition, 1968.

[9] N. Alon and J. Spencer, *The Probabilistic Method*, John Wiley & Sons, 2000.

[10] J.H. Wilkinson, *The Algebraic Eigenvalue Problem*, Oxford Science Publications, 1965.

[11] G.H. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins University Press, 1989.

[12] A. Sinclair, *Algorithms for Random Generation and Counting: A Markov Chain Approach*, Springer-Verlag, 1997.

[13] A. Broder and A. Karlin, "Bounds on the cover time," *J. Theoretical Probab., 2:101-120*, 1989.

[14] D. Aldous and J. Fill, "Reversible markov chains and random walks on graphs," *Monograph, http://stat-www.berkeley.edu/users/aldous/book.html*, 2002.

[15] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomikns, and J. Wiener, "Graph structure in the web," *9th International World Wide Web Conference (WWW9)/Computer Networks*, vol. 33, no. 1-6, pp. 309–320, 2000.

[16] S. Dill, R. Kumar, K. McCurley, S. Rajagopolan, D. Sivakumar, and A. Tomkins, "Self-similarity in the web," in *International Conference on Very Large Data Bases*, Rome, 2001, pp. 69–78.

[17] Gopal Pandurangan, Prabhakar Raghavan, and Eli Upfal, "Building low-diameter p2p networks," in *IEEE Symposium on Foundations of Computer Science*, 2001, pp. 492–499.

[18] C. Gkantsidis, M. Mihail, and A. Saberi, "Conductance and congestion in power law graphs," in *Proc. SigMetrics*. ACM, 2003.

[19] Fan Chung, Lincoln Lu, and Van Vu, "The spectra of random graphs with given expected degrees," *Proceedings of National Academy of Sciences*, vol. 100, no. 11, pp. 6313–6318, 2003.

[20] M. Mihail, C.H. Papadimitriou, and A. Saberi, "On certain connectivity properties of internet graphs," *FOCS*, 2003, to appear.

[21] Andy Oram, Ed., *Peer-to-Peer: Harnessing the Power of Disruptive Technologies*, O'Reilly, first edition, 2001.

[22] David Gibson, Jon M. Kleinberg, and Prabhakar Raghavan, "Inferring web communities from link topology," *UK Conference on Hypertext*, 1998.

[23] C. Gkantsidis, M. Mihail, and E. Zegura, "Spectral analysis of internet topologies," in *Proc. Infocom*. IEEE, 2003.

[24] R. Kumar, S. Rajagopalan, D. Sivakumar, and A. Tomkins, "Crawling the web for emerging cyber-communities," in *WWW8/Computer Networks*, 1999, vol. 31, pp. 1481–1493.

[25] Tian Bu and Don Towsley, "On distinguishing between internet power law topology generators," in *Proc. Infocom*. IEEE, 2002.

[26] N. Leibowitz, M. Ripeanu, and A. Wierzbicki, "Deconstructing the kazaa network," in *Proc. WIAPP*. IEEE, June 2003, pp. 112–120.

[27] Limewire.org, "Snapshots of the gnutella network," *http://crawler.limewire.org/data.html*, 2002.

[28] M. Faloutsos, P. Faloutsos, and C. Faloutsos, "On power-law relationships of the internet topology," in *Proc. SigComm*. ACM, 1999.

[29] M. Jovanovic, F. Annexstein, and K. Berman, "Modeling peer-to-peer network topologies through small-world models and power laws," in *IX Telecommunications Forum*, 2001, http://www.ececs.uc.edu/~mjovanov/Research/telfor.pdf.

[30] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, and S. Shenker, "Making gnutella-like p2p systems scalable," in *Proc. SIGCOMM*. ACM, 2003, to appear.

[31] Albert-László Barabasi and Réka Albert, "Emergence of scaling in random networks," *Science*, vol. 286, pp. 509–512, 1999.