Adding structure to unstructured peer-to-peer networks: the role of overlay topology

Shashidhar Merugu Sridhar Srinivasan Ellen Zegura
College of Computing,
Georgia Institute of Technology,
Atlanta, GA 30032
{merugu, sridhar, ewz}@cc.gatech.edu

***Abstract*— Our work examines the role of overlay topology on the performance of unstructured peer-to-peer systems. We focus on two metrics of performance: (a) search protocol performance, a local gain perceived directly by a user of the system and (b) utilization of the network, a global property that is of interest to network service providers. We present a class of overlay topologies based on distance between a node and its neighbors. We show, by simulation, that a particular topology instance of this class where every node has many close neighbors and few random neighbors exhibits better properties than other examined instances. In this overlay topology, the chances of locating files are high and the nodes where these files are found are, on average, close to the query source. This improvement in search protocol performance is achieved while decreasing the traffic load on the links in the underlying network. We propose a simple greedy algorithm to construct such topologies where each node operates independently and in a decentralized manner to select its neighbors.**

I. Introduction

Sharing of files is the most dominant application in use on current peer-to-peer networks [1]. Unstructured and decentralized systems (e.g., BearShare, LimeWire based on Gnutella [2], Kazaa based on FastTrack [3]) are extremely popular and attractive for their simplicity. We focus on these decentralized and unstructured peer-to-peer networks to support file searching; our goal is to examine the role of overlay topologies on the performance of such systems.

Gnutella and its family of protocols [4] use a very simple *scoped-flooding search* mechanism to locate files. A search query is propagated to all neighbors from the source node. The query is replicated and forwarded by each intermediate node to all its neighbors, up to a system-defined maximum number of overlay hops from the source. Every intermediate node compares its local contents with the requested file in the search query and responds to the query source on a match. Scoped-flooding is costly when scaled to large numbers and can only find files within the search radius. However, it is very simple and amenable to a dynamic real-world peer-to-peer network. Unlike structured peer-to-peer networks (e.g. Chord [5], CAN [6], Pastry [7]), unstructured peer-to-peer networks do not have any association between the content and location where it is stored, thereby eliminating the complexity of maintaining such an association in a dynamic scenario [8]. Without an association between content and location, a node does not have any information about which other nodes can best be able to resolve a query. In this case of a "blind" search, scoped-flooding reaches many search candidates as well as limits the distance travelled by a search query.

The success of a scoped-flooding search, both in terms of *quantity* of candidates searched, and the *quality* of results returned, is strongly affected by the topology of the peer-to-peer network. For example, in an overlay topology that is a k-ary tree rooted at the source of the query, a scoped-flooding search up to m hops would examine k^m candidates. On the other hand, if the source and its neighbors form a clique in the overlay topology, many nodes are revisited and hence the number of unique search candidates is lower. The quality of a result can be measured as the distance (e.g., latency, available bandwidth) in the underlying network from the hit node to the query source. A *close* result may be able to better serve the file than a far node. In an overlay topology that is constructed unaware of the underlying network, as is the case with Gnutella topologies [9], [10], the search query hits may be far from the source, even though they are within the scoped-flooding search radius on the overlay topology. The overlay topology also determines the amount of traffic load (search queries/results) on the underlying network. For example, when most of the overlay links cross Autonomous System (AS) boundaries, there is a proportional amount of inter-AS traffic [10]. While the study of the characteristics of overlay topologies is very important to the performance of peer-to-peer networks, their effect on scoped-flooding search and the traffic load on links in the underlying network has received little attention.

In this paper, we present a class of overlay topologies for decentralized and unstructured peer-to-peer networks, taking inspiration from the recent work on small-worlds [11], [12]. Small-world topologies exhibit *clustering* with some random edges. It has been observed that the small-world structure is pervasive and arises "without-design" or "naturally" in many practical systems such as the World Wide Web [13], [14]. Overlays, on the other hand, provide an opportunity to *design structure*. We seek the advantages of designing overlays with a small-world structure. In our model of overlay topologies, a node's links to its neighbors are divided into two categories: *short* links and *long* links. The short links connect close nodes and the long links connect nodes chosen randomly. The fraction of links that are short, called the *proximity factor* (α), is a key design parameter that controls the properties of the resul-

tant overlay topology. When $\alpha = 0$, every link is randomly chosen, and the overlay topology is a *random graph*. On the other hand when $\alpha = 1$, every link connects to a close neighbor, and the overlay topology looks more well-ordered (e.g., grid-like). Different values of α let us span the spectrum of this class of overlay topologies. We show, using simulation, that overlay topologies with an invariant of *many close neighbors and few random neighbors* at each node exhibit the following properties: *(a)* the underlying network distance to search candidates increases with the overlay radius as we progress in the scoped flooding search; *(b)* the count of search candidates is fairly high; and *(c)* the system returns close results. We also show that the traffic load on links of the underlying network is better for these overlay topologies than random overlay topologies.

Given our results showing that these overlay topologies perform well, we turn to the practical question of constructing such topologies in a dynamic peer-to-peer environment. We propose a method of topology construction where each node operates independently and in a decentralized manner to select its own neighbors. Neighbors are selected from a pool of candidates using a simple greedy algorithm based on local information. In a practical implementation, we foresee short links connecting nodes belonging to the same AS where possible, while the long links cross AS boundaries. Given the dynamic conditions of a peer-to-peer network, nodes periodically evaluate the distance to their neighbors and replace them if necessary to maintain the invariant ratio of short and long links.

The remainder of the paper is organized as follows. We study the class of overlay topologies in Section II and present properties of peer-to-peer systems that are important for their performance in Section III. In Section IV, we present our evaluation methodology and simulation results. We propose an algorithm for overlay topology construction in Section V. Section VI has related work followed by a discussion in Section VII. We conclude with a summary in Section VIII.

II. Overlay Topologies

In this section, we present a class of overlay topologies for decentralized and unstructured peer-to-peer systems. We discuss how different instances of this class of topologies can be realized by varying a characteristic parameter.

One of the key purposes of characterizing overlay topologies is to create a simple framework for an empirical analysis. Our choice for a model for the overlay topologies is inspired by the generic lattice network model for small-worlds [15]. Since the hosts on the Internet and their locations with respect to each other do not resemble grid points on a lattice, the generic lattice network model is not directly applicable. However, we adopt its salient feature: characterization of a network topology based on a typical node and its distance to its neighbors. Our classification of overlay topologies for peer-to-peer systems based on the distance of a node to its neighbors has some advantages. It is based only on a node's local view and thus helps us in design of a distributed algorithm to construct such topologies. Also, the notion of locality in the underlying network is preserved by incorporating distance to neighbors.

In our class of topologies, there are two kinds of links: *short-links* and *long-links*. Every node in the system selects some *close* nodes to be its short-link neighbors and some nodes *at random* to be its long-link neighbors. An appropriate metric for distance (e.g., latency) in the underlying network defines the *closeness* of neighbors. While the short-link neighbors are carefully chosen to be close to the node, the long-link neighbors, on the other hand, being chosen randomly, are generally far from the node. We make an implicit assumption that the system has a large enough population of nodes that are spread across the underlying network ensuring the existence of close neighbors. Crawls of Gnutella peer-to-peer topologies [16], [9] and live-count of active Gnutella peers [17] support this focus on a large population.

The *proximity factor* (α) of a node, defined as the ratio of the number of short links to the total number of links, is a design parameter that governs the overall structure of the topology. A node with degree d has αd short links and $(1-\alpha)d$ long links. α takes values from 0 to 1, inclusive: $\alpha = 0$ corresponds to all long-links and $\alpha = 1$ corresponds to all short-links. Different values of α let us span the spectrum of this class of overlay topologies. Figure 1 is an illustration of a node's links in four instances of this class of topologies.

In an instance of the class of topologies with $\alpha = 0$, all the neighbors of a node are randomly chosen, and this results in a *random graph*. It is well-known that random graph topologies have a low diameter and are fairly robust in terms of connectivity [18]. The average distance between neighbors, in this all long-links case, is typically the mean node-to-node distance in the underlying network. As α increases, there is less "randomness" and more "order" in the topology. The "order" in the topology comes from the increasing number of short-links. Short-links "induce" more short-links. Intuitively, if A and B are short-link neighbors of C, then chances are that A and B are also close to each other and could be short-link neighbors themselves. This possibility of a node's neighbors being neighbors themselves introduces "clusters" in the topology. Clustering, a measure of "cliqueness" of a neighborhood, is formally defined as the fraction of allowable edges that occur among the set of neighbors of a node [13]. As α increases, the number of short-links increase and the topology is increasingly clustered. The instance of topology with $\alpha = 1$ corresponds to all short-links and we expect the topology to contain multiple clusters and to be possibly disconnected. However, an instance of the topology with many short-links but few long-links, i.e., with an α between 0.5 and 1.0, is expected to exhibit both the properties of a random graph and a clustered graph. We call such instances "small-world-like" overlay topologies. These small-world-like topologies contain multiple clusters formed by the many short-links, but their random long-links connect across clusters and keep the graph connected as well as lower its diameter.

In the following sections, we observe how the variation in the proximity factor changes the properties of the overlay topology. We highlight important metrics for performance improvement in the next section and later seek a suitable instance of overlay topologies by an experimental evaluation.

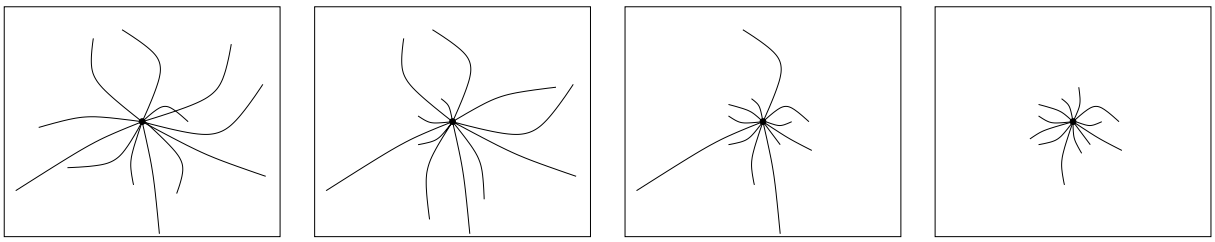

Fig. 1. An illustration of different topologies in the class of overlay topologies obtained by varying the proximity factor α. Each of the four pictures shows the links associated with a typical node in the overlay topology. The length of each link is indicative of distance to that neighbor in the underlying network.

III. Desirable Properties

In this section, we present metrics that are used to study the effect of overlay topologies on the performance of peer-to-peer systems. We focus on two properties of peer-to-peer systems: *(a) search protocol performance*, a local gain perceived directly by a user of the system and *(b) utilization of the network*, a global property that is of interest to network service providers. We also discuss the relationship between these desirable properties and the overlay topologies.

A. Search Space

The size of the search space around a query source node is measured by the number of unique candidate nodes visited. The success of a scoped-flooding search often depends on the size of search space since the more search candidates, the greater the chances of finding the requested file. The factors that determine the size of search space include search radius and the structure of the topology. There is a trade-off between the value of the search radius and the usage of system resources for propagating a search query. A larger search radius implies the search query lives longer in the network and travels farther from its source. Also, the overall completion time of the search, depending on response, if any, from the farthest nodes increases with search radius. Current unstructured peer-to-peer systems usually define a system-wide constant search radius to limit the resource usage. Of the structural properties of a topology, the amount of "clustering" determines the size of search space. A scoped-flooding search on a highly clustered topology revisits the same nodes multiple times lowering the count of unique search candidates. On the other hand, a scoped-flooding search on a topology with similar number of nodes and edges but with lower clustering might find more unique search candidates.

It is also desirable that the overlay topology be always connected and robust from sudden departures of nodes that are quite common in a dynamic peer-to-peer network. Connectivity of the topology ensures that scoped-flooding search does not terminate early and that all nodes up to the search radius are explored.

B. Underlying Network Distance to Search Results

After completion of the search process when the search results are returned, the query source has a choice of selecting one of the hit nodes to serve the file. This file transfer depends on the distance between the query source and the chosen hit node in the underlying network. Though the query source and hit node are relatively close in the overlay, it is not necessary that these two nodes are close in the underlying network. If the overlay topology is *aware* of the underlying network, the scoped-flooding search mechanism can find close hit nodes. We define our notion of *awareness* as correspondence between the search traversal on the overlay topology to a progress in distance on the underlying network. Consider a node p on the overlay topology. Let D_i denote the average distance in the underlying network to all nodes that are exactly i hops away on the overlay topology. We define the *stride* at i^{th} step away from p to be the term $(D_i - D_{i-1})$. Consider a plot of D_i as a function of i. The number of overlay hops is on the x-axis, and the average distance in underlying topology is on the y-axis, and draw a curve with points: (i, D_i). The slope of the curve between $(i-1)$ and i is the stride at i^{th} step. A positive stride at all steps means that the distance in underlying network increases as we reach out farther nodes on the overlay topology.

C. Network Traffic

It has been observed that the most of the packets contributed by the unstructured peer-to-peer systems are either the search queries/replies or pings/pongs that exchange neighbor state information [10]. Although these packets are small in size, their large numbers make up a high load on the links in underlying network. A benefit of an overlay topology being aware of the underlying network is an improvement in the utilization of links in underlying network. One of the most common definitions of traffic load in overlay networks is the notion of *stress* (e.g., [19], [20], [21]). Stress [19] of a link in underlying network is defined as the number of logical links (on the overlay topology) whose mapped paths include the underlying link. If we assume that one unit volume of traffic is exchanged between every pair of neighbors on the overlay topology, the value of stress of an underlying link defines the cumulative traffic carried by the underlying link. Though the values of stress on links in underlying network estimate the traffic load, perhaps the most important links where the measure of stress has a lot of significance are the links connecting different ASes. Various reasons, mainly economic, contribute to the significance of

traffic load on the inter-AS links. It is considered beneficial by the network service providers to reduce the traffic exchanged on these inter-AS links. Clustering in a topology often conforms very well with the AS level division of the underlying network. When a node and its neighbors that make up a cluster in an overlay topology belong to the same AS, most of their search traffic and pings/pongs remain within the AS boundaries.

In summary, we desire overlay topologies that satisfy a dual objective of improving the performance of scoped-flooding search protocol and minimizing the traffic load on the links of the underlying network. To realize these goals, we seek "good" structural properties on the overlay to have *(a)* a large number of unique search candidates and *(b)* an awareness of overlay topology with the underlying network. An overlay topology with structural properties similar to that of a random graph gives large number of unique search candidates, however it does not "fit" well with underlying network. On the other hand, an overlay topology with high clustering conforms well with the underlying network, but has a lower number of unique search candidates. We aim to find a suitable balance in the structural properties of overlay topologies for these two goals of producing a higher count of search candidates and being underlying network-aware.

IV. Evaluation

In this section, we analyze, by simulation, how the desirable properties described in Section III vary as we vary the proximity factor (α) in our class of overlay topologies.

A. Methodology

1) Simulation Parameters: We used simulation to evaluate overlay topologies along the different dimensions of desirable properties. A network topology comprising of 5152 routers was generated using the Transit-Stub graph model from the GT-ITM topology generator [22]. The link latencies are assigned values according to the type of the link, using a uniform distribution on the following ranges: [15ms, 25ms] for intra-transit domain links, [3ms, 7ms] for transit-stub links, [1ms, 3ms] for intra-stub links. In each experiment, end-hosts are attached uniformly at random to the stub routers of the underlying network topology. These end-hosts participate in the overlay network. All our experiments have 4096 end-hosts (overlay nodes) by default. All overlay nodes have at least 3 and at most 4 neighbors[1]. In order to simulate scoped-flooding search for files, 50 unique files with varying popularity are introduced into the system. Each file has multiple copies stored at different locations chosen at random. The number of copies of a file are proportional to their popularity. The count of file copies is assumed to follow Zipf distribution with 2000 copies for the most popular file and 40 copies for the least popular file. The queries that search for these files are also initiated at random hosts on the overlay topology. Again the number of queries for a file is assumed to be proportional to its popularity. Each query follows the scoped-flooding search protocol up to a maximum of 4 overlay hops from the query source[2]. We keep track of the number of search candidates seen by each search query and average distance to these search candidates in the underlying network. In addition to measuring quantities related to scoped-flooding search, we also measure the diameter and average path length of the overlay topology and the stress on the links in the underlying network. Each experiment is repeated multiple times with different seeds to remove any bias in random number generation that is used in multiple stages of simulation (e.g., placement of end-hosts on the underlying network, copies and queries of files on end-hosts).

2) Overlay Topologies: We constructed four different instances of the class of overlay topologies described in Section II. These four instances of topologies are compared along different dimensions of the desirable properties described in Section III. The instances of topologies have proximity factors of $\{0.0, 0.33, 0.66, 1.0\}$. With a minimum degree of 3 at each node, these four values of proximity factor correspond to four combinations of numbers of short and long links: $(0, 3)$, $(1, 2)$, $(2, 1)$ and $(3, 0)$. While the long-link neighbors are randomly chosen from the node population, global information about underlying network distances to all nodes is used in deciding the short-link neighbors[3].

An overlay topology instance with $\alpha = 0.0$ ("all-long-links"), corresponds to the case where neighbors of every node are randomly chosen. Current unstructured peer-to-peer systems use a neighbor selection scheme that is similar to this all-long-links instance of overlay topologies. When a node joins the peer-to-peer system, it contacts a *gateway* that returns a set of randomly picked seed-neighbors from its cache of known nodes. The neighbor relationship is maintained as long as the neighbors are active and have resources. When a neighbor departs, the node finds a random replacement from the candidate set of nodes that it learns during its life-time. There is no evidence of locality on the underlying network in either choice of initial neighborhood or replacement nodes. Since the neighbor selection is random, the overlay topology of current peer-to-peer systems resembles that of a random graph. Hence this kind of all-long-links topology is representative of the real-world peer-to-peer topologies. At the other end of the spectrum, every node has close neighbors in the "all-short-links" topology with $\alpha = 1.0$. The instance of $\alpha = 0.66$ is a topology where every node has more short links than long links and has similarities to a small-world graph. We also refer to this "many-short-links" instance as "small-world-like" overlay topologies.

[1] Implementations of Gnutella protocol (including Limewire [23], GTK-Gnutella [24], Gnucleus [25]) allow users to configure the number of neighbors. Typical recommended values for node degree are 4 and 5. We ran our experiments with the degree ranges of $[4, 6]$ inclusive and the results are similar to what we present here. One key difference though is in the diameter of the overlay topology that decreases with increasing degree.

[2] Though early versions of Gnutella protocol specification suggested a TTL of 7, most popular Gnutella servents including Limewire [23] use a default TTL value of 4 to limit system resource usage. http:// core.limewire.org/ source/ browse/ core/ com/ limegroup/ gnutella/ SettingsManager.java (v1.245)

[3] Since we are only interested in studying the various kinds of topologies, we believe that this "oracle" approach of constructing overlays is acceptable in our simulation environment. However, implementing a particular choice of overlay topology in a practical environment would have to work without this assumption of global knowledge of distance to all nodes. We address this practical issue of topology construction later in Section V.

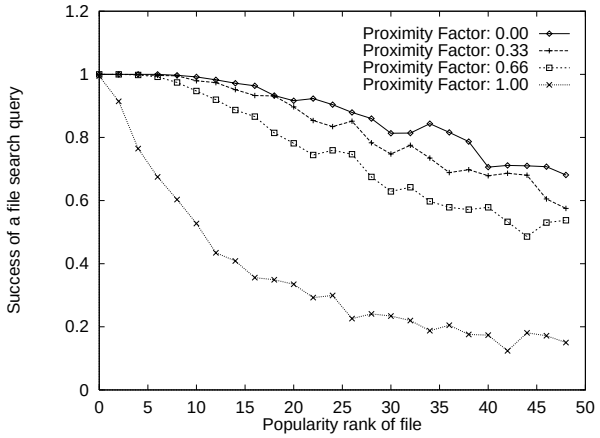

Fig. 2. Scoped-Flooding Search Query Success Rate

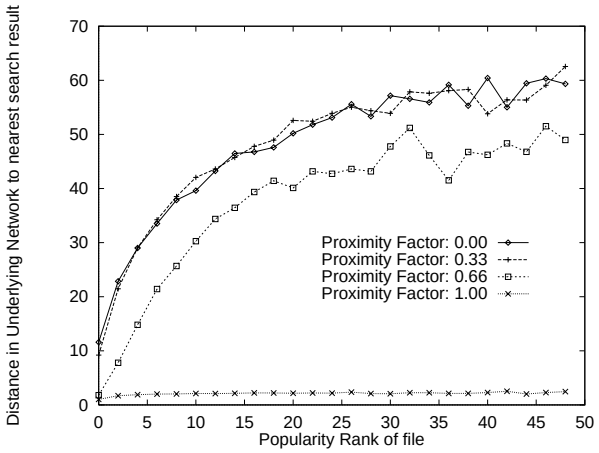

Fig. 4. Underlying network distance to nearest search result

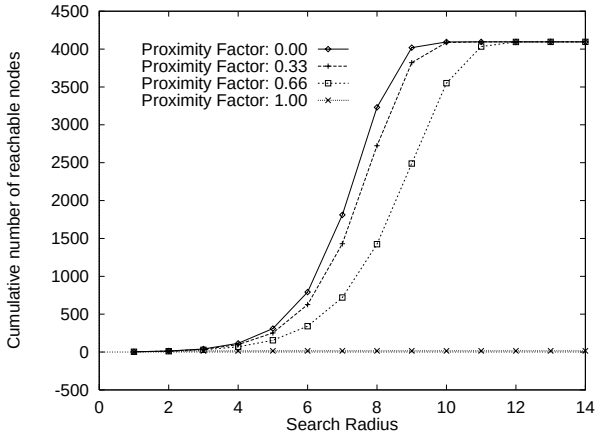

Fig. 3. Count of reachable nodes with increasing search radius

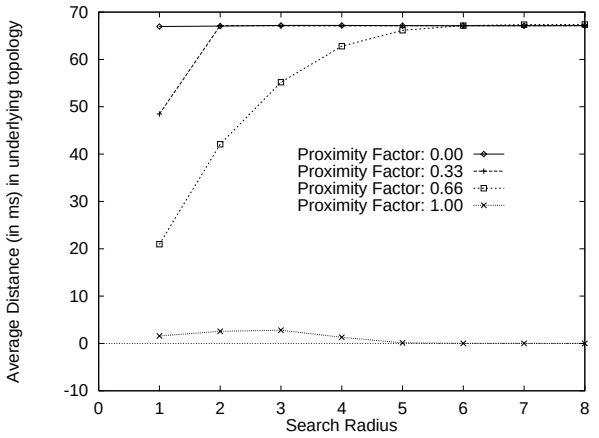

Fig. 5. Variation of average underlying network distance to nodes at a given radius

B. Simulation Results

1) Success of Search: In this experiment, we studied the success rate of scoped-flooding searches for each type of topology. A query for a file is "successful" when it finds at least one match for the requested file within the search radius. Success rate of a file is defined as the ratio of number of successful queries to the total number of queries issued for that file throughout the system. Though the number of queries we simulated per file is proportional to the popularity of that file, "success rate" is a normalized quantity across all the files. In Figure 2, we plot the success rate of queries for files with different popularity for each instance of overlay topologies. All topologies have a high success rate for more popular files because of the large number of copies for these files. However as the popularity decreases, the number of copies of files in the system decreases and the success rate for all the topologies goes down. One key observation is that the success rate for the all-short-links topology (with $\alpha = 1.0$) drops rapidly, compared to the success rate for the other three values of the proximity factor. We also note that the success rate for small-world-like topology is fairly close to that of the all-long-links topology.

The effects that we observe in Figure 2 follow directly from Figure 3. In Figure 3, we plot the cumulative number of unique nodes that are reachable as we increase the search radius of the scoped-flooding scheme. The all-long-links topology ($\alpha = 0$) reaches more nodes at each radius than all the other three topologies; but with a maximum search radius of 4 that is used in our simulation, the number of unique nodes reached by these three topologies ($\alpha = \{0, 0.33, 0.66\}$) is very similar. The small-world-like topology can access a fairly large number of nodes because each long link allows the query to reach more nodes outside its local region. The similarity in number of search candidates results in similar success rate that we observed in Figure 2 for these three topologies. Also, the low success rate of the all-short-links topology in Figure 2 is due to the small (almost constant) number of nodes that are accessed at increasing search radii. This topology causes nodes to form clusters and makes the graph disconnected. Table I lists the diameter and average path length of the four kinds of overlay topologies. Note that the diameter and the average path length of the small-world-like topology ($\alpha = 0.66$) is comparable to the more random topologies ($\alpha = \{0.0, 0.33\}$). On the other hand, the all-short-links topology is disconnected in all our multiple trials and the values for diameter and average path length that are reported in this table are for the connected components of the topology.

Proximity Factor	0.0	0.33	0.66	1.0
Overlay Diameter	11	12	14	6
Avg. Path Length	7	7	8	2

TABLE I

PATH PROPERTIES OF OVERLAY TOPOLOGY

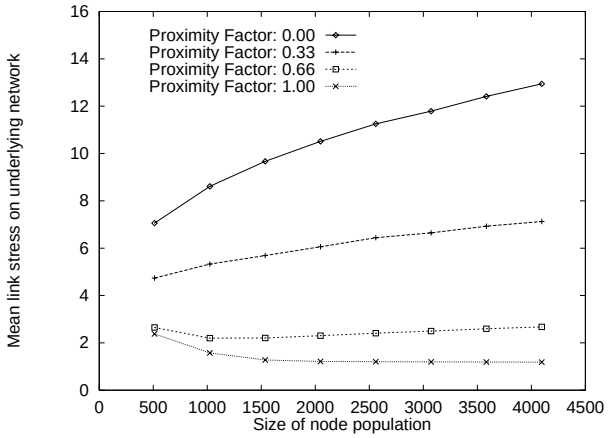

Fig. 6. Variation of mean stress on links of the underlying network as the size of node population increases for different kinds of topologies

2) Distance to search results:

Of the multiple file locations returned by a scoped-flooding search, the nearest one in the underlying network may be able to better serve the file than other locations. We study the variation of this distance to the nearest search result node for the four kinds of topologies in Figure 4. The underlying network distance to nearest search result is plotted as a function of file popularity. Each point is an average over all the queries issued for that file. The nearest search result for small-world-like topologies ($\alpha = 0.66$) is closer than the nearest result for topologies with $\alpha = 0.0$ and $\alpha = 0.33$. Even though the all-short-links topology shows the smallest distance, most of the queries are failures. The reason for these observations follows directly from Figure 5 where we plot the average distance in the underlying network to nodes located at increasing values of the search radius. This metric called "stride", as defined in Section III-B, measures "awareness" of an overlay topology with the underlying network. The stride for all-long-links topology ($\alpha = 0$) is flat (zero) as all the links are randomly chosen and hence the average distance to nodes is independent of how far they are on the overlay from the source node. It can also be clearly seen that the neighborhood of nodes in small-world-like topology is closer in the underlying network (a positive stride), leading to the search results being found closer in the underlying network. On the other hand, in the all-short-links topology, all the nodes at increasing radii are very near the source and exhibit a tendency to form clusters. We also notice that, on average, after a small number of overlay hops, the all-short-links topology is disconnected.

3) Stress: To show the effect of traffic load by the various topologies on the underlying network, we plot the mean stress

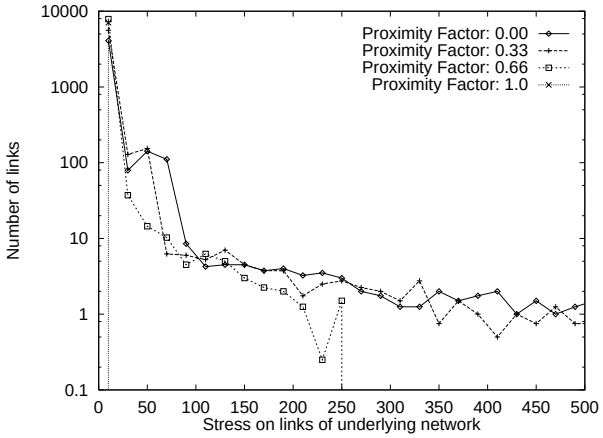

Fig. 7. Distribution of stress on links of the underlying network

of links in the underlying network as the size of the population increases in Figure 6. The all-short-links overlay topology has extremely low mean stress because for every node, its set of neighbors is selected from nodes that are close to it and the links to those nodes are not shared by any other pair of nodes, resulting in low stress. In the case of all-long-links ($\alpha = 0.0$) topology, a node's neighbors are picked from the entire set of nodes, leading to overlay links that span the underlying network, causing underlying links to carry multiple overlay links. The many-long-links ($\alpha = 0.33$) topology is in between the two extremes. The graph shows that the amount of randomness in choosing the neighbors increases the mean stress of the topology. The small-world-like ($\alpha = 0.66$) topology, by virtue of having many short links, has a low mean stress like that of the all-short-links topology. Some of the disproportionate increase in stress of more random topologies ($\alpha = \{0.0, 0.33\}$) as the population size increases, is potentially due to the use of transit-stub graphs for our simulations. The links connecting stubs to transit nodes could possibly form bottlenecks pushing up the mean stress of a topology. To investigate this, we plot the frequency distribution of the stress on underlying links in Figure 7. It can be clearly seen that random topologies have many more links with high stress than the all-short-links and small-world-like topologies on the same underlying network.

C. Summary of results

A summary of the observed properties of the four kinds of topologies is given in Table II. As expected of random graphs, the all-long-links topology has a low diameter, reaches out to many candidates in a scoped-flooding search and is always connected. However, it does not utilize the links of underlying network well and the stride is almost zero for these topologies. On the other end of the spectrum, the all-short-links topology behaves like a well-ordered graph with higher diameter, and better utilization of the links in underlying network. However, the search space is quite small and the topology is disconnected most often. In between these two ends of the spectrum, but closer to the case of all-short-links, we observe that the topologies, with many short links and few long links, have desirable properties. They not only have low diameter, large search space

and are always connected like an all-long-links topology, but are also aware of the underlying network and can utilize the links better like an all-short-links topology.

V. Overlay Topology Construction

We have shown in the previous section that overlay topologies with many close neighbors and few random neighbors at each node exhibit desirable properties. Now we consider how to practically construct such overlay topologies in a peer-to-peer environment. In this section, we explain our incremental greedy algorithm for overlay topology construction. Our algorithm design goals include the following:

1) **Scalable** with the number of participant nodes.
2) **Distributed** in execution using only local information available at a node. We do not expect to use either special infrastructure or any cooperation of a central agent for the construction or maintenance of the topology.
3) **Resilient** to dynamic arrivals and departures of nodes in a peer-to-peer network.

We propose an operation called `Adapt` that is executed by nodes in the peer-to-peer system. By executing an `Adapt` operation, a node selects "better" neighbors according to its local view of the system. Each node in the system has the following parameters: *(1)* the number d of neighbors; *(2)* the proximity factor α that determines the neighborhood ratio of short and long links; *(3)* a *adapt-threshold* γ of distance to determine whether another adapt operation is necessary; and *(4)* an *adapt-epoch* δ value to schedule a periodic adapt operation. The life-time of a node on the peer-to-peer system is divided into two phases: a join phase and a maintenance phase. In the join phase, a new node, say X, adapts repeatedly until its short-link neighbors are at a "satisfactory" distance. Let D_i denote the average distance in underlying network to the short-link neighbors of X after i^{th} invocation of `Adapt`. X adapts until: $|D_i - D_{i-1}| < \gamma$. When this condition is met, say, at time t_0, we say X has "settled" in its neighborhood. Later, in the maintenance phase, X schedules a periodic `Adapt` operation at every adapt-epoch δ time units, i.e., at $\{(t_0 + \delta), (t_0 + 2\delta), \ldots\}$. An `Adapt` operation is also triggered when one or more neighbors depart.

A description of the `Adapt` procedure in pseudo-code is given in Figure 8. The three input parameters to this procedure are the node-ID (u), the number of short links (s) and the number of long links (l). s and l are computed according to the node parameters d and α where $s = \alpha d$ and $l = (1 - \alpha)d$. In the first two steps, node u builds a candidate list C of potential neighbors from its current local neighborhood. These candidates include current neighbors N of u and neighbors of current neighbors, i.e., all nodes seen in a "depth-first-traversal" starting from u up to a depth of two hops. Node u measures distances to each of these candidates $v \in C$ from itself. A new set of neighbors N' is formed by selecting the s "closest" candidates according to distance rank and picking l candidates at random from the rest. Thus node u acts in a greedy fashion to improve its local neighborhood.

One of the key features of the `Adapt` procedure is its simplicity. `Adapt` is similar in spirit to the simple scoped-flooding search of the unstructured peer-to-peer systems. Both schemes scout their immediate neighborhood, but for different reasons: one for closer neighbors, the other for locating files. `Adapt` is decentralized and does not make use of any central authority or any infrastructure. It is based only on a local view of the node and hence can be easily deployed on current peer-to-peer systems. The number of messages exchanged in each `Adapt` operation are $O(\mathcal{D}^2)$ where $\mathcal{D}$ is the max-degree of a node. When the max-degree $\mathcal{D}$ of nodes in the peer-to-peer system is a constant, the computational complexity of these `Adapt` operations is bounded by a constant. The triggers for an `Adapt` operation, both based on an adapt-epoch and adapt-threshold, make this scheme amenable to dynamic arrivals and departures of nodes of a peer-to-peer system.

```
Adapt(u, s, l) { // u: node, s short links and l long links
    // get neighbors of u
    N ← u.nbrList
    // build candidate list
    C ← N ∪ {∪_{v∈N} v.nbrList}
    // compute distance to each candidate
    For each node v in C {
        D[v] ← distance(u, v)
    }
    // sort the distance array
    sort(D)
    // find 's' closest and 'l' random candidates
    N' ← closest(D, s) ∪ random(D, l)
    // assign new neighbors
    u.nbrList ← N'
}
```

Fig. 8. Pseudo-code of `Adapt` procedure

A. Evaluation

We have conducted preliminary evaluation of the join phase of our proposed method for constructing overlay topologies. We have also examined the question of partial deployment, where only a fraction of the node population observe the invariant ratio of many close neighbors and few random neighbors. We use the same evaluation framework and methodology as described in Section IV-A. We present both these results in this section.

1) Join phase: There are two metrics to evaluate the cost of the join phase: *(1)* the number of `Adapt` operations performed before the termination condition, and *(2)* the variation in distance to neighbors with multiple `Adapt` operations. Fewer `Adapt` operations mean a rapid and short join phase. We also seek a low final distance to neighbors at the end of the join phase. In this experiment, we randomly chose 256 locations in the underlying network for a new node. This new node has three short links and one long link. It starts with seed-neighbors selected at random from the node population. This random choice of seed-neighbors is repeated 16 times. In all, we conducted $256 \times 16 = 4096$ runs of this experiment to evaluate the join phase. A statistics computation on the numbers of `Adapt` operations performed in each join phase reveal a mean of 3.5 and

Topology kind → Property ↓	All long links	Many long links	Many short links	All short links
Network Diameter	Low	Low	**Low**	High
Connectedness	Connected	Connected	**Connected**	Disconnected
Search Space	Large	Large	**Large**	Small
Stride	Zero	Zero	**Positive**	Positive
Network Utilization	Poor	Poor	**Good**	Good

TABLE II

A SUMMARY OF PROPERTIES OF THE FOUR KINDS OF TOPOLOGIES.

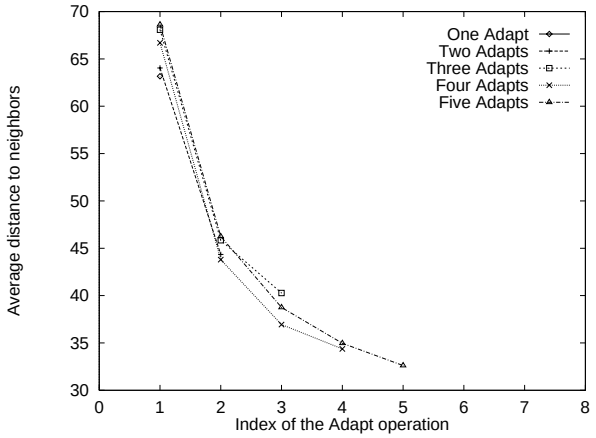

Fig. 9. Decrease in average underlying network distance to neighbors with multiple Adapt operations

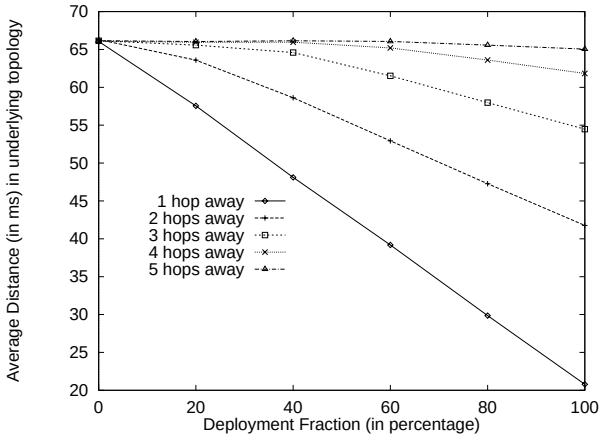

Fig. 10. Average distance to nodes under partial deployment

a standard deviation of 1.55. Figure 9 shows the reduction in the average distance to neighbors with each `Adapt` operation. Each curve corresponds to the set of runs that terminated after specified numbers of `Adapt` operations. We can note from the figure that the average distance to neighbors decreases with `Adapt` operations. However, the join phase terminates when the final average distance to neighbors is 38.5ms, on average over all the runs, even though closer nodes exist in the system. It is possible to refine our simple greedy algorithm to find such closer nodes. For example, instead of a depth of two, one could think of a deeper traversal so that a node can explore more candidates for closer neighbors. But this improvement comes at a price of increase in measurement of distance to a larger set of candidates, thereby increasing the message complexity of this algorithm.

2) Partial Deployment: In this experiment, we vary the percentage of nodes that use proximity factor of $\alpha = 0.66$ corresponding to the many-short-links topology instance. The rest of the nodes in the system use a proximity factor of $\alpha = 0.0$ corresponding to the all-long-links topology instance. In Figure 10, we plot the average distance to nodes at different radii and Figure 11 shows the variation in mean stress on the links of the underlying network for different sizes of node population as the fraction of deployment is varied. Each of these figures show gradual improvement in the average distance and mean stress as the fraction of the node population with $\alpha = 0.66$ is varied from 0% to 100%.

VI. RELATED WORK

Ripeanu et al. [9] looked at the projection of Gnutella logical overlay on the underlying network-layer topology. Their experiment on counting the number of logical edges that stayed within an AS and those that crossed AS boundaries revealed a significant "mismatch" between the logical overlay and the projection on the underlying network. A second study by Saroiu et al. [16], also based on a crawl of Gnutella topology, confirmed that it is highly robust to partitions indicating its similarity to a well-connected random graph. A third study by Jovanović et al. [26] evaluated clustering coefficients and average path lengths of Gnutella virtual topologies. However, this study is based on a crawl performed when Gnutella was in its nascent stages (late 2000) and since there is no known record or a model for growth of Gnutella, it is not clear how to extrapolate the values of clustering and average path lengths to current versions of Gnutella topologies. The large scale and the complexity of current Gnutella servents make it very difficult to capture an accurate picture of Gnutella network today. There is an ongoing

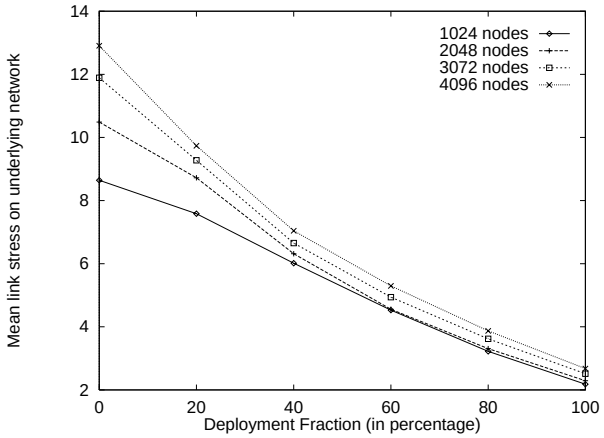

Fig. 11. Mean stress on links of the underlying network for partial deployment

effort in the open-software community [27] to crawl and visualize the Gnutella network topology.

On another front, there have been many efforts on construction of topology aware overlay networks. Distributed binning [28] is based on the idea that if two nodes perceive similar (in terms of rank or order) distances to well-known Internet landmarks, then they must be near each other. Another proposal called "beaconing" [29] uses an algorithm based on triangulation heuristics to identify nearby nodes by querying a set of "beacons". This approach is not scalable as every beacon needs to keep track of all nodes in the system along with their respective distances from it. Addressing the scalability issue in beaconing, an alternative based on hierarchical trees called Tiers [30] is proposed. These approaches to find close nodes are complementary to our work. They can be integrated with our algorithm to find the short-link neighbors at a node.

Among other considerations for constructing topologies of unstructured peer-to-peer networks, a recent proposal [31] suggests "interest-based shortcuts" to link peers that have similarity in shared content. Search queries are forwarded on the shortcut links before exploring the regular links. Their simulation results show considerable improvement in object location using this simple idea of interest based locality. It is possible to think of a composite metric that combines both distance in underlying topology that we use in our work with the interest based locality to select neighbors.

A recent work, very similar in spirit to ours, combines the idea of small-worlds with peer-to-peer networks. In this work [32], the authors propose a new cache replacement algorithm for routing tables to improve location of data items in Freenet [33]. Information about neighbors is cached in a node's routing table if they contain data items that are "close" in key space to node's own key. Under the invariant condition, each node would point to many neighbors that store "close" data items and few neighbors that store "randomly far" data items. There are two main differences between this work and ours. First, this distance measure is purely over the key space and doesn't consider the distance in underlying network. We expect that this key space distance metric can blend with our distance metric of the underlying network. Second, Freenet design includes the computation of routing tables and query forwarding based on file identifier look-ups. There are no explicit routing tables in Gnutella as their calculation and maintenance is complex due to the large size and dynamics of peers. Hence Gnutella, which relies only on scoped-flooding search, may not benefit from optimizations on the key space.

VII. Discussion

While we firmly believe that small-world-like overlay topologies are promising for unstructured peer-to-peer systems, there are several issues that need further investigation. Our class of overlay topologies assumed that all nodes are uniform in terms of their degree. This uniformity assumption led us to a very simple characterization of the class of topologies and studying their properties became easy. However, in practice, nodes of a peer-to-peer system are known to be heterogenous in their resources and quite possibly have a wide range of degrees. Many properties of the topology, including diameter, connectivity, are affected by this diversity in degrees of nodes. It is very hard to characterize these complex topologies, let alone study the application driven metrics such as size of search space or traffic load on the underlying network links when using these complex overlay topologies.

A very interesting question is the relationship between the dynamics in the population of nodes and maintenance of overlay topology invariants. There is a cost associated with imposing any "structure" to the overlay topology, for example, the cost of finding close neighbors. Under rapid arrivals and departures of nodes, it is likely that the cost of maintaining the invariant ratio of short and long links could exceed the benefit from such topologies. We plan to explore different optimization criteria for these special cases of high dynamics in node population.

We had suggested an adapt-threshold for the terminating condition of the join phase and an adapt-epoch to schedule periodic `Adapt` operations in the maintenance phase. Each `Adapt` operation includes some processing at the node and exchange of messages (measurement probes) to estimate distance to candidate neighbors. A "good" choice of values for adapt-threshold and adapt-epoch would balance the cost of `Adapt` and its corresponding gain. We need further experience with using `Adapt` to estimate good values for these parameters. We could start with a simple AS level definition for adapt-threshold such that a neighbor that is within the same AS as the node "meets" the adapt-threshold. Periodic `Adapt` operations could be replaced with "lazy" `Adapt` operations that are performed only when needed.

VIII. Summary

Overlay topology plays a fundamental role in the performance of an unstructured peer-to-peer network. In this paper, we have presented a class of overlay topologies and their characteristic parameter called proximity factor. Proximity factor relates to the underlying network distance between a node and its neighbors. Different instances of this class of overlay topologies are realized by varying the proximity factor. These topology instances are examined, by simulation, along

two performance-related dimensions: *(a)* search protocol performance and *(b)* utilization of links in the underlying network. Our simulation results show that in a particular "small-world-like" topology instance of this class where every node has many close neighbors and few random neighbors, *(1)* the chances of locating files are high and *(2)* the nodes where these files are found are, on average, close to the query source. This improvement in search protocol performance is achieved while *(3)* decreasing the traffic load on the links in the underlying network.

To demonstrate the feasibility of constructing such "small-world-like" overlay topologies in a practical peer-to-peer environment, we have proposed a simple greedy algorithm called `Adapt`. A node executing `Adapt` operates independently and in a decentralized manner to select its neighbors. Preliminary evaluation suggests that `Adapt` finds closer neighbors and our approach is incrementally deployable.

REFERENCES

[1] A. Oram, Ed., *Peer-to-Peer: Harnessing the Power of Disruptive Technologies*, O'Reilly, 2001.

[2] "http://gnutella.wego.com," .

[3] "http://www.fasttrack.nu," .

[4] "http://rfc-gnutella.sourceforge.net," .

[5] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications," in *ACM SIGCOMM*, 2001.

[6] S. Ratnaswamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A Scalable Content Addressable Network," in *ACM SIGCOMM*, 2001.

[7] A. Rowstron and P. Druschel, "Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems," in *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, 2001.

[8] S. Ratnaswamy, S. Shenker, and I. Stoica, "Routing Algorithms for DHTs: Some Open Questions," in *First International Workshop on Peer-to-Peer Systems*, March 2002.

[9] M. Ripeanu, I. Foster, and A. Iamnitchi, "Mapping the Gnutella Network: Properties of Large Scale Peer-to-Peer Systems and Implications for System Design," *IEEE Journal on Internet Computing, Special Issue on Peer-to-peer Networking*, 2002.

[10] S. Sen and J. Wang, "Analyzing peer-to-peer traffic across large networks," in *Internet Measurement Workshop*, 2002.

[11] D. J. Watts, Ed., *Small Worlds: The Dynamics of Networks between Order and Randomness*, Princeton University Press, 1999.

[12] J. Kleinberg, "Navigation in a small world," in *Nature*, 2000.

[13] D. J. Watts and S. H. Strogatz, "Collective Dynamics of Small-World Networks," *Nature*, 1998.

[14] R. Albert, H. Jeong, and A. L. Barabási, "The Diameter of the World-Wide Web," *Nature*, 1999.

[15] J. Kleinberg, "The Small-World Phenomenon:An Algorithmic Perspective," in *ACM Symposium on Theory of Computing*, 2000.

[16] S. Saroiu, K. Gummadi, and S. Gribble, "A Measurement Study of Peer-to-Peer File Sharing Systems," in *Multimedia Conferencing and Networking*, Jan 2002.

[17] "Gnutella network hosts: http://www.limewire.com/current_size.html," .

[18] B. Bollobás, Ed., *Random Graphs*, Cambridge University Press, 2nd edition, 2001.

[19] Y. Chu, S. G. Rao, S. Seshan, and H. Zhang, "A Case for End System Multicast," *IEEE Journal on Selected Areas in Communication, Special Issue on Networking Support for Multicast*, 2002.

[20] S. Banerjee, S. Bhattacharjee, and C. Komareddy, "Scalable Application Layer Multicast," in *ACM SIGCOMM*, 2002.

[21] M. Kwon and S. Fahmy, "Topology-Aware Overlay Networks for Group Communication," in *NOSSDAV*, 2002.

[22] E. Zegura, K. Calvert, and M. J. Donahoo, "A Quantitative Comparison of Graph-based Models for Internet Topology," *IEEE/ACM Transactions on Networking*, vol. 5, no. 6, Dec 1997.

[23] "http://www.limewire.com," .

[24] "http://gtk-gnutella.sourceforge.net," .

[25] "http://www.gnucleus.com," .

[26] M. A. Jovanović, F. S. Annexstein, and K. A. Berman, "Modeling peer-to-peer network topologies through "small-world" models and power laws," *IX Telecommunications Forum, TELFOR*, 2001.

[27] "Crawler project: http://crawler.limewire.org," .

[28] S. Ratnaswamy, M. Handley, R. Karp, and S. Shenker, "Topologically-Aware Overlay Construction and Server Selection," in *Infocom*, 2002.

[29] N. Shankar, C. Komareddy, and S. Bhattacharjee, "Finding Close Friends over the Internet," in *International Conference on Network Protocols*, 2001.

[30] S. Banerjee, C. Komareddy, and S. Bhattacharjee, "Scalable Peer Finding on the Internet," in *Global Internet Symposium, Globecom*, 2002.

[31] K. Sripanidkulchai, B. Maggs, and H. Zhang, "Efficient Content Location Using Interest-Based Locality in Peer-to-Peer Systems," in *Infocom*, 2003.

[32] H. Zhang, A. Goel, and R. Govindan, "Using the Small-World Model to Improve Freenet Performance," in *Infocom*, 2002.

[33] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong, "Freenet: A Distributed Anonymous Information Storage and Retrieval System," *Lecture Notes in Computer Science*, 2001.