On Constructing Internet-Scale P2P Information Retrieval Systems

D. Zeinalipour-Yazti, V. Kalogeraki and D. Gunopulos

Department of Computer Science & Engineering
University of California - Riverside,
Riverside CA 92521, USA
{zeinalipour,vana,dg}@cs.ucr.edu

Abstract. We initiate a study on the effect of the network topology on the performance of Peer-to-Peer (P2P) information retrieval systems. The emerging P2P model has become a very powerful and attractive paradigm for developing Internet-scale systems for sharing resources, including files, or documents. We show that the performance of Information Retrieval algorithms can be significantly improved through the use of fully distributed topologically aware overlay network construction techniques. Our empirical results, using the Peerware middleware infrastructure, show that the approach we propose is both efficient and practical.

1 Introduction

In the last few years, the new emerging Peer-to-Peer (P2P) model has become a very powerful and attractive paradigm for developing Internet-scale file systems [18, 12, 20, 21, 23] and sharing resources (i.e., CPU cycles, memory, storage space, network bandwidth) [22] over large scale geographical areas. This is achieved by constructing an overlay network of many nodes (peers) built on top of heterogeneous operating systems and networks. Overlays are flexible and deployable approaches that allow users to perform distributed operations such as information retrieval [9, 30] without modifying the underlying physical network.

The first wave of P2P systems implement *unstructured* P2P overlays [12] in which no global structure or knowledge is maintained. To search for data or resources, messages are sent over multiple hops from one peer to another with each peer responding to queries for information it has stored locally. *Structured* P2P overlays [20, 21, 23] implement a distributed hash table data structure in which every data item can be located within a small number of hops, at the expense of keeping some state information locally at the nodes. Recently more efficient query routing techniques based on routing indices [8], heuristics [28] and caching [30] were proposed.

However, an important problem that these systems have not fully considered is how the heterogeneity of the underlying infrastructure affects the performance of the information retrieval algorithms/systems. The P2P infrastructure can encompass resources with different processing and communication capabilities, located across different geographical areas. As a result, retrieving information over

Fig. 1. Typical end-to-end delays from Riverside to sites around the world. It is important to minimize the network delays (or distances) in Internet-Scale P2P systems.

Internet-scale environments is subject to greater variations due to unpredictable communication latencies, excessive resource consumption and changing resource availability.

Figure 1 shows typical end-to-end delays from Riverside to different sites around the world. We can see that hosts at different locations have different latencies which usually dependent on a number of factors such as geographic distances, network traffic, load at respective sites and others. By exploiting network proximity, the overlay application can alleviate the overheads that are usually imposed by the operation of the Peer-to-Peer network. It is worth mentioning that exploiting the physical network structure can be useful in a number of other P2P applications such as content distribution networks, spam detection networks and others.

P2P systems are very effective mechanisms to share and store documents, because their decentralized nature allows easy additions, updates, large storage, and offers fault-tolerant properties through the use of replication and caching. However, a system for storing large amounts of data should also provide efficient search mechanisms, and the decentralized nature of the unstructured P2P networks hinders the use or the maintenance of the indexing structures traditionally used in Information Retrieval. So the effective use of P2P systems for document storage depends on new efficient and distributed solutions to the problem of finding the documents one is looking for.

In the paper we focus on *keyword search*, that is, we aim to find the documents that contain a given set of query terms. There is a lot of recent work on improving keyword search in unstructured P2P networks (section 2 provides an overview). A common theme in this work is the use of the number of messages as a metric of the performance of the technique. While this is justified when the

algorithm is network-agnostic and does not use the characteristics of the network to improve the search, we believe that taking advantage of this knowledge can significantly improve the performance of Information Retrieval and allow us to design techniques that make the problem practical in Internet-scale systems.

Our Contribution: In this paper we initiate a study on the design of fully distributed P2P information retrieval techniques that are topologically aware and can take advantage of the network characteristics to optimize the efficiency of the search. We consider and evaluate the impact of the use of topologically aware overlay network constructions on the accuracy and the performance of currently proposed fully distributed P2P information retrieval techniques. Although the necessity of topologically-aware overlays has been widely addressed in the context of structured overlays [4, 19, 27, 31], it hasn't received the same attention in the context of unstructured overlay networks. More specifically:
(i) We discuss and evaluate the performance of information retrieval algorithms over topologically-aware Internet-scale overlays. We consider both *agnostic* techniques that do not maintain any knowledge of the data distribution in the network (BFS and RBFS), as well as techniques that collect past statistics (>RES, ISM). We describe the Random and BinSL overlay construction techniques and describe the advantages and disadvantages of the BinSL technique for the P2P Information Retrieval problem.
(ii) We study the impact of the overlay construction techniques on the information retrieval algorithms using our *Peerware* infrastructure. Our objective is to improve the latency, while maintaining the accuracy of the results. We note here that our results show that the use of topologically-aware overlays minimizes network delays while maintaining high recall rates and a low number of messages.

The remainder of the paper is organized as follows: In section 2 we outline search techniques that have been proposed for efficient information retrieval in unstructured P2P networks. In section 3 we describe the Random and BinSL overlay construction techniques and describe their advantages and disadvantages. Section 4 describes our experimental methodology and our middleware infrastructure. In section 5 we present our experimental evaluation and section 6 concludes the paper.

2 Search Techniques for Unstructured P2P Networks

In this section we provide a brief overview of techniques and algorithms that can be used to perform content-based searches in P2P system. The techniques do not use any global knowledge, thus they are completely decentralized and scale well with the size of the network. We consider a network of n nodes (peers). We assume that D_u is the set of documents that are stored in peer u. We assume that each document d is characterized by a sequence of keywords, and let $s(d)$ be the (unordered) set of keywords in d. Given a query q, itself a set of keywords, the result of the query should be the *answer set*

$\{(d, u)|u$ is a peer and q $\subset s(d)$ and $d \in D_u\}$, that is, the documents in the network that include the keywords in q.

Agnostic techniques: Breadth First Search (BFS) and Random BFS (RBFS): BFS is a technique widely used in P2P file sharing applications, such as Gnutella [12]. BFS sacrifices performance and network utilization for the sake of simplicity. It works as follows : A node v generates a `Query` message q when it wants to search for contents located on other peers. v propagates q to all its neighbor peers. When a peer u receives a `Query` request, it first propagates q further by again along its neighbors (except the sender), and then searches its local repository for relevant matches. If some node w has a match, w generates a `QueryHit` message to transmit the result. `QueryHit` messages are sent along the same path that carried the incoming `Query` messages. The disadvantage of BFS is that a query is consuming excessive network and processing resources because a query is propagated along all links. In order to avoid flooding the network with queries, as the network might be arbitrary large, each query is associated with a time-to-live (TTL) field which determines the maximum number of hops that a given query should be forwarded.

In [15] we propose and evaluate the *Random Breadth-First-Search (BFS)* technique. RBFS improves over the naive BFS approach by allowing each node to forward the search request to only a fraction of its peers. This fraction can be selected at random and is a parameter to the mechanism (in our experiments we used a fraction of 0.5, so that a peer propagates the request to half its peers, selected at random). This technique uses fewer messages than BFS, however it may miss large segments of the network since it is random and may not choose a particular link that could propagate the query to such segments.

The Most Results in the Past Heuristic (>RES): In [28], Yang et al., present a technique where each node forwards a query to some of its peers based on some aggregated statistics. The authors compare a number of query routing heuristics and mention that the *Most Results in Past (>RES)* heuristic has the best satisfaction performance. A query is defined to be *satisfied* if Z, for some constant Z, or more results are returned. In >RES a peer u forwards a search message to the k peers which returned the most results for the last 10 queries.

The technique is similar to the Intelligent Search Mechanism we describe below, but uses simpler information about the peers, and is optimized to find Z documents efficiently (for a fixed Z) rather than finding as many documents as possible. The nature of >RES allows it to explore the larger network segments (which usually also contain the most results) and the most stable neighbors (the peers that have routed back many queryhits), but it doesn't manage to explore the nodes which contain content related to the query. We therefore characterize >RES a *quantitative* rather than *qualitative* approach.

The Intelligent Search Mechanism (ISM): In [15] we propose the Intelligent Search Mechanism (ISM) which is a fast and efficient mechanism for information retrieval in unstructured P2P networks.

Keys to improving the speed and efficiency of the information retrieval mechanism is to minimize the communication costs, that is, the number of messages sent between the peers, and to minimize the number of peers that are queried for each search request. To achieve this, a peer estimates for each query, which of its peers are more likely to reply to this query, and propagates the query message to those peers only.

The Intelligent Search mechanism consists of two components that run locally in each peer:

The *Profile Mechanism* is used to maintain the T most recent queries and the corresponding queryhits along with the number of results. Once the repository is full, the node uses the Least Recently Used (LRU) replacement policy to keep the most recent queries.

The *RelevanceRank (RR)* function is used by a node P_l to perform an online ranking of its neighbors in order to determine to which ones to forward a query q. To compute the ranking of each peer P_i, P_l compares q to all queries in the profiling structure, for which there is a queryhit, and calculates $RR_{P_l}(P_i, q)$ as follows:

$$RR(peer_i, q) = \sum_{q_j = \text{"Queries answered by } peer_i\text{"}} sim(q_j, q)^{\alpha} * results(q_j)$$

The deployed distance metric $Qsim$ is the cosine similarity[1] and $S(P_i, q_j)$ is the number of results returned by P_i for a query in the profiling structure q_j. RR allows us to rank higher the peers that returned more results. α allows us to add more weight to the most similar queries. For example, when α is large then the query with the largest similarity $Qsim(q_j, q)$ dominates the formula. If we set $\alpha = 1$ all queries are equally counted, while setting $\alpha = 0$ allows us to count only the number of results returned by each peer (essentially, the >RES heuristic). In the experiments we forward the query to the half best neighbors, plus to a random neighbor to brake out of potential cycles. More details about the RR function can be found in [30].

ISM works well in environments which exhibit strong degrees of query locality and where peers hold some specialized knowledge. Our study on the Gnutella network shows that it exhibits a strong degree of query locality [29].

Other Distributed Techniques and Algorithms: In the previous subsections we described techniques that do not index the shared content. We now describe other proposed techniques which deploy indexing, intelligent data and key placement as well as advanced techniques from the Information Retrieval field in order to improve keyword search performance. It is important to mention that many of the described techniques comes with certain overheads, therefore their applicability might be limited in very large environments.

There is a lot of work on centralized systems, however, these are not directly relevant to our problem as these systems assume some centralized component that assists in the indexing procedure. In [8], Crespo et al., present a hybrid technique where each peer builds indices using aggregate information on the contents of the documents of its peers. In the *PlanetP* [9] system, participating nodes build a global inverted index which is partially constructed by each node.

The framework is based on bloom filters, which capture the index of some node, and which are randomly gossiped across the community. In a different approach, the pSearch [24] system explores semantic spaces by using advanced techniques from the Information Retrieval field. It uses the Vector Space Model (VSM) and Latent Semantic Indexing (LSI) to generate a semantic space which is then distributed on top of a CAN [20] structured P2P overlay.

In the Random Walker model, which is presented in [17], each node forwards a query message by selecting a random neighbor and the query message is called a *walker*. This model however doesn't use any explicit technique to guide the query to the most relevant content. In *APS* [25] each node deploys a local index, which captures the relative probability of each neighbor to be chosen as the next hop for some future request. The main difference with Random Walkers is that in APS a node utilizes feedback from previous searches to probabilistically guide future walkers.

Distributed file indexing systems such as CAN[20] and Chord[23] allow peers to perform efficient searches using object identifiers rather than keywords. These systems, usually referred as *Structured Overlays* or *Distributed Hash Tables (DHT)*, use a specific structure with some hashing scheme that allows peers to perform object lookup operations getting in return the address of the node storing the object. A disadvantage of DHTs is that they consider only the problem of searching for keys, and thus cannot perform content-based retrieval. Recent work in [11] shows that content-based query resolution is feasible in DHT systems if these are using *Rendezvous Points (RP)*. More specifically the framework proposes the registration of the content (i.e. attribute-value pairs that describe the content) at *RP*s. Queries might then be routed, using Chord, to a predefined set of *RPs* which consequently resolve the query. Finally Freenet [7], is another distributed information storage and retrieval system that uses instead an intelligent *Depth-First-Search (DFS)* mechanism to locate the object keys in the system. The advantage of DFS search is that a small set of peers can be queried quickly and efficiently; however by its nature it can take a long time if we want to find all the results to a query.

So far we have seen different information retrieval techniques and algorithms for Peer-to-Peer environments. In the next section we will discuss how the construction of the overlay network affects the performance of the discussed algorithms and why it is important to optimize such a parameter.

3 Overlay Topologies for Efficient Network Utilization

In this section we discuss two distributed overlay construction techniques that can be deployed in the context of unstructured overlay networks. Let $G = (V, E)$ be an overlay topology, with a vertex set $V = \{1, 2, ..., n\}$ and an edge set E, which represents the overlay connections between the vertices in V. Assume that a user, is connected to some vertex v and that it uses one of the search techniques outlined in the previous section in order to search for content in G. Then his query is expected to form a spanning tree T which spans over the subgraph G'

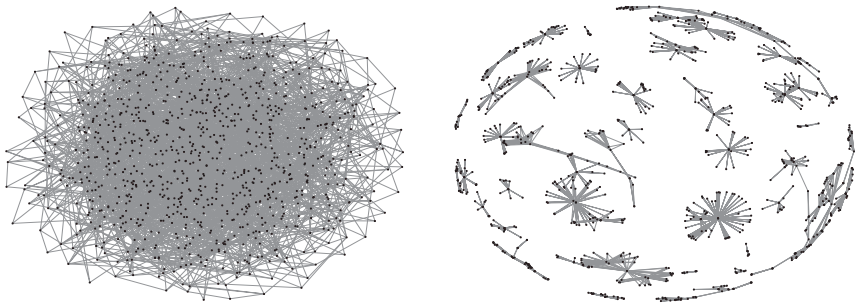

Fig. 2. Visualization of a Connected (Random) and Disconnected (SS) graph of 1000 peers (degree=6) using the Fruchterman-Reignold visualization model in Pajek [2]. Random and BinSL topologies have the advantage that they remain connected while *SS* topologies get disconnected because each node greedily selects other close-by nodes.

($G' \subset G$). The main goal of an overlay construction technique is to minimize the **Aggregate Delay (Δ_T)** which is the sum of the delays w associated with each edge in the tree T, more formally defined as following: $\Delta_T = \sum_{\forall \epsilon \in T} w(\epsilon)$

It is important to notice that the delay cost associated with each edge might be different for each direction between two nodes v_i and v_j (i.e. $delay(v_i \rightarrow v_j) \neq delay(v_j \rightarrow v_i)$). This happens because packets on the Internet may follow different itineraries or because the upstream and downstream bandwidth of a node might greatly vary (e.g. Cable/ADSL Modem Users). Another interesting point is that the construction of an optimal overlay is known to be NP-complete [10] therefore the following popular algorithms are based on heuristics.

Random Topology: In this algorithm, each vertex v_i selects its d neighbors by randomly choosing d other vertices. This is the algorithm deployed in most current P2P networks such as [12] and its main advantages are that it is simple, does not require any knowledge on the distances, and leads to connected topologies if the degree $d > log_2 n$ [3].

BinSL Topology: In [19], Ratnasamy et al. propose the Short-Long (SL) overlay construction technique. SL alleviates the network unawareness of the Random Topology in the following way: Each vertex v_i, selects its d neighbors by picking the $d/2$ nodes in the system that have the shortest latency to itself (these connections are called *short links*) and then selects another $d/2$ vertices at random (these connections are called *long links*). Therefore SL requires the nxn IP-latency matrix in order to find the latencies between the various node pairs. The intuition behind this algorithm is that the $d/2$ connections to "close-by" nodes will result in well-connected clusters of nearby nodes, while the random links serve to keep the different clusters interconnected and the overall graph connected. It is important to mention that by only selecting the shortest latency

nodes will in most cases result in disconnected graph topologies. This can be observed in the visualization of figure 2 where we visualize a random and a short (SS) topology of 1000 peers.

Although the SL construction technique works well in practice, it is limited by the fact that some node in the system needs to know the "physical" distances between all node pairs (i.e. an nxn IP-latency adjacency matrix). In practice such centralized architectures don't scale well, are expensive and are vulnerable to denial of service attacks. In order to overcome the global knowledge requirement of the SL algorithm, Ratnasamy et al. propose the *BinSL* topology construction technique [19], which is a distributed adaptation of the SL algorithm. Since the adjacency-matrix of IP latencies is not available in a distributed setting, *BinSL* deploys the notion of *distributed binning* in order to approximate these latencies. More specifically each node calculates the round-trip-time (RTT) from itself and k well-known *landmarks* $\{l_1 l_2..l_k\}$ on the Internet. The numeric ordering of the latencies represents the "bin" the node belongs to. Latencies are then further classified into *level* ranges. For instance if the latencies are divided into 3 levels then; *level 0* accounts for latencies in the range [0,100), *level 1* for the range [100,200) and *level 2* for all other latencies. The level vector is then augmented to the landmark ordering of a node yielding a string of the type "$l_2 l_3 l_1 : 012$". It is expected that nodes belonging to the same bin will be topologically close to each other although *false positives* are possible, that is, some nodes do belong to the same "bin" although they are not topologically close to each other. We will investigate the accuracy of the binning scheme in the experimental section.

Other Topologically-Aware Construction techniques: Recently an approach to create resilient unstructured overlays with small diameters was proposed in [26]. In the proposed algorithm a node selects from a set of k nodes, r nodes at random ($r \subset k$) and then finds from the rest $f=k$-r nodes the ones that have the largest degree. This algorithm results in networks with power-law distributions of node degrees differentiating from Random and BinSL in which we have a uniform distribution.

Topologically-aware overlays have also been addressed in the context of *Structured* P2P overlays in [4, 19, 27, 31]. These systems however rely on some hashing scheme which allows nodes to quickly send messages to some destination node. Although structured overlays are of particular importance in applications such as decentralized web caches [14], they are not appropriate for content-based retrieval systems [9, 30] and large-scale systems with transient user populations [5].

Application-layer multicast systems such as Narada [6] initially construct a richer connected graph (mesh) and then use some optimization algorithm to generate a mesh that has certain performance properties. As part of the mesh quality improvement algorithm, Narada nodes randomly probe each other and calculate the perceived gain in utility. BinSL is simpler and cheaper in terms of messages. It is furthermore designated for larger groups of members, which might leave and join in an ad-hoc manner.

4 Experimental Evaluation Methodology

Our experimental evaluation focuses on three parameters: (i) the **aggregate tree delay (Δ_T)** which is a metric of network efficiency for a given query that spans in the sub-graph G', (ii) the **recall rate**, that is, the fraction of documents each of the search mechanisms retrieves, and (iii) the **overhead** of the techniques, that is, the number of messages that are consumed in order to find the results. As the baseline of comparison we used the results retrieved by querying the collection in a centralized setting (i.e. as a corpus of documents) which therefore returns all relevant documents. We chose to implement the algorithms that require only local knowledge (i.e. BFS, RBFS, >RES and ISM) over Random and BinSL topologies of the same size and degree.

The TREC Dataset: We use two series of experiments which are based on the *TREC-LATimes* dataset which is a document collection that consists of randomly selected articles that appeared on the LA Times newswire from 1989 to 1990. The size of this dataset is 470MB and it contains approximately 132,000 articles. These articles were horizontally partitioned into 1000 xml documents each of which was subsequently indexed using the Lucene [16] IR API. These indexes, which are disk-based, allow the efficient querying of text-based sources using many IR features. We then generate Random and BinSL topologies of 1000 peers in which each peer shares one or more of the 1000 documents (see figure 3a). We use this scheme in order to provide some degree of article replication. We don't use the *"qrels"* relevance judgments, since the compared algorithms don't attempt to address the issue of precise document retrieval. We will refer to these peers as the *TREC-LATimes Peerware*.

For the evaluation of the TREC-LATimes corpus we will use, as indicated by NIST, the TREC topics 300-450. One problem with the provided 150 queries is that the query term frequency is very low and most terms are presented only once. This is not a realistic assumption since studies on real P2P networks (e.g. [29]) indicate that there is a high locality of query terms. Therefore we used the 150 queries to derive the **TREC50x2** dataset, which consists of a set a ="50 randomly sampled queries out of the initial 150 topics". We then generated a list b of another 50 queries which are randomly sampled out of a. *TREC50x2* is then the queries in a and b randomly shuffled and the distribution of query terms can be viewed in figure 3b.

Simulating Network Distances: Evaluating distances in network topologies requires a dataset in which the IP latencies are not synthetic. We didn't chose to use a real dataset of ≈300,000 IPs found in the Gnutella network [29], as obtaining the distances among the different hosts was practically not feasible. We therefore chose to base our experiments on the measurements of the *Active Measurement Project (AMP)* [13], at the *National Laboratory for Applied Network (NLAR)*. AMP deploys a number of monitors distributed along 130 sites to actively monitor the Internet topology. AMP monitors ping and traceroute each other at regular intervals and report the results back to the project servers. Most

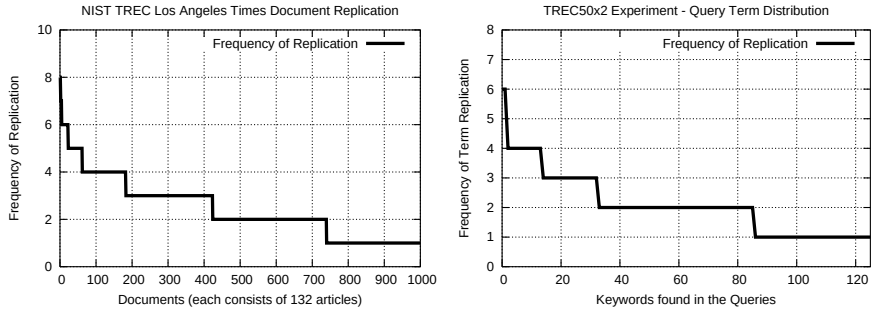

Fig. 3. a) Data Replication scheme for the TREC-LATimes dataset, **b) Query Term Frequency** distributions for the TREC50x2 queryset.

of the current 130 monitors currently reside in the U.S with a few exceptions of some other International sites.

In our experiments we use an AMP 1.8 GB snapshot of traces obtained on the 30th of January 2003. The set includes data from 117 monitors out of which we extracted the 89 monitors which could be reversed DNS (i.e. given their IP we obtained a DNS name). We then construct the nxn IP-latency matrix (for all n=89 physical nodes), that contains the latency among all monitors. Since all 89 hosts are located at different domains, we chose to incorporate some degree of host replication per domain. Our study in [29] shows that hosts in a real overlay network, such as Gnutella, exhibit this characteristic. More specifically we randomly replicate each host $[1..k]$ times. In our experiments we set $k = 24$ which generated distances for all 1000 nodes in the TREC-LATimes Peerware.

Peerware Simulation Infrastructure: In order to benchmark the efficiency of the information retrieval algorithms, we have implemented *Peerware*[1], a distributed middleware infrastructure which allows us to benchmark different query routing algorithms over large-scale P2P systems. We use Peerware to build a decentralized newspaper network which is organized as a network of 1000 nodes. Our experiments are performed on a network of 75 workstations (each hosting a number of nodes), each of which has an AMD Athlon 800MHz-1.4GHz processor with memories varying from 256MB-1GB RAM running Mandrake Linux 8.0 (kernel 2.4.3-20) all interconnected with a 10/100 LAN. Peerware is written entirely in Java and comes along with an extensive set of UNIX shell scripts that allow the easy deployment and administration of the system.

Peerware consists of three components: (i) *graphGen* which pre-compiles network topologies and configuration files for the various nodes participating in a given experiment, (ii) *dataPeer* which is a P2P client that is able to answer to boolean queries from its local xml repository using the Lucene IR Engine [16], and (iii) *searchPeer* which is a P2P client that performs queries and harvests

[1] Details about the Peerware infrastructure can be found in [30].

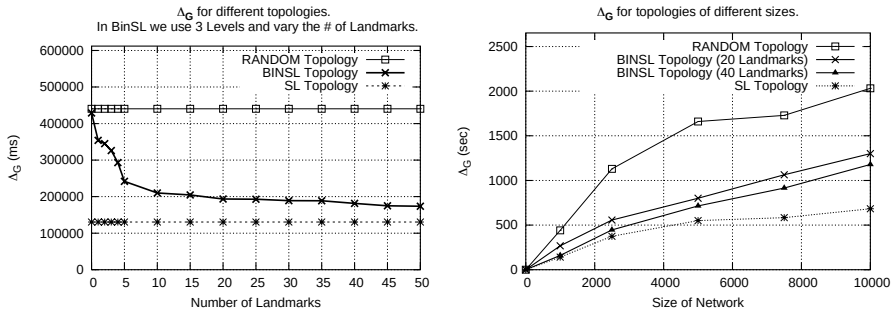

Fig. 4. a) The rate of *false positives* of the Binning scheme is a function of how many landmarks are used. b) Scalability of the Binning Scheme.

answers back from a Peerware network. Launching a Peerware of 1000 nodes can be done in approximately 10-20 seconds while querying the same network can be performed in around 250ms-1500ms.

The Discarded Message Problem: We define the DMP problem in the following way: Node P_k receives some query q with TTL_1 at time t_1. P_k first checks if it has forwarded the same query (identified by GUID) in the past. If yes, it will immediately discard the message in order to avoid forwarding the message several times. If not, it will decrease TTL_1=TTL_1-1 and forward q to some of P_k's peers. Now what happens if node P_k receives the same query q with some TTL_2, where $TTL_2 > TTL_1$ at some time t_2, where $t_2 > t_1$? Most of the commercial P2P clients will discard q. The result is that a query reaches fewer nodes than expected. We fix the problem by allowing the TTL_2 message to proceed. Of course there is some redundancy which will add up in the "number of messages" graph (approximately 30% in the experiments).

5 Experiments

In this section we describe a series of experiments that attempt to investigate the effect of the Random and BinSL overlay topology structure on the recall rate and the messaging of the various information retrieval search techniques that were described in section 2. We are particularly interested in investigate if the BinSL topology can indeed minimize the aggregate network delay without sacrificing the recall rate.

Efficiency of Landmarks in BinSL: The first experiment, we attempt to find the right number of landmarks, as this plays an important role on how small the Δ_T becomes in a fixed size network. By using more landmarks, the number of *false positives* also decreases as we have fewer collisions in the landmark ordering codes of hosts that are not topologically close to each other. In figure 4a, we calculate the sum of the delays w associated with all edges in the respective graphs

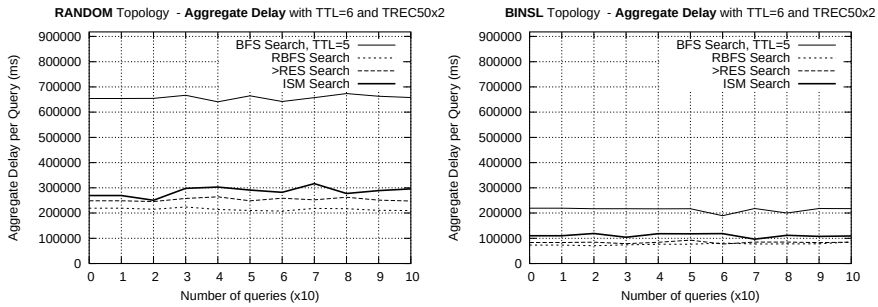

Fig. 5. Aggregate Delay for the evaluation of the TREC50x2 queryset using the Random (left) and BinSL (right) topology.

G (1000 peers each with an average degree of 6). This sum is more formally defined as $\Delta_G = \sum_{\forall \epsilon \in G} w(\epsilon)$ and we use this metric, instead of the Aggregate Delay Δ_T, as it is independent of the deployed search technique. In BinSL, we first randomly sample out of the original network the set of landmarks.[2] The figure indicates that by using no landmarks, the BinSL topology is essentially a Random topology. This happens because a node selects all its connections at random which makes Δ_G of the Random and BinSL topologies identical. By adding a few landmarks (i.e. 1-10), Δ_G decreases substantially, after which point Δ_G decreases at a lower rate. Therefore by selecting an arbitrary large number of landmarks may not be very efficient as each landmark probing comes with an additional network cost and because Δ_G may not significantly drop.

Although figure 4a shows that by using 20 landmarks might be satisfactory for a network of 1000 nodes, in practice the network size might not be known a priori. In figure 4b, we plot the Δ_G parameter for networks of different sizes. The figure indicates that the Random Topology does not scale very well with respect to the Δ_G parameter. By using BinSL and 20 landmarks on the other hand, the Δ_G parameter decreases by 46% from what the Random topology uses, while using 40 landmarks drops Δ_G by 54%. We can see that although we doubled the number of landmarks the Δ_G parameter improved by only 8%. The picture also shows that the lower bound provided by SL is on average 66% less than what the random topology requires, but SL is not feasible in practice as it requires global knowledge. For the rest of the paper we set the number of landmarks to 20.

Minimizing Network Delays: In our second experiment, we investigate if we can minimize the Aggregate Delay Δ_T of a query that spans in the subgraph G', while retaining high recall rates and low messaging. In the BFS case, we configure each query messages with a TTL parameter of five since this technique is consuming extraordinary amounts of messages. With this setting, query mes-

[2] In a real setting, peers would have a predefined list of well chosen landmarks (i.e. globally spread HTTP or DNS servers).

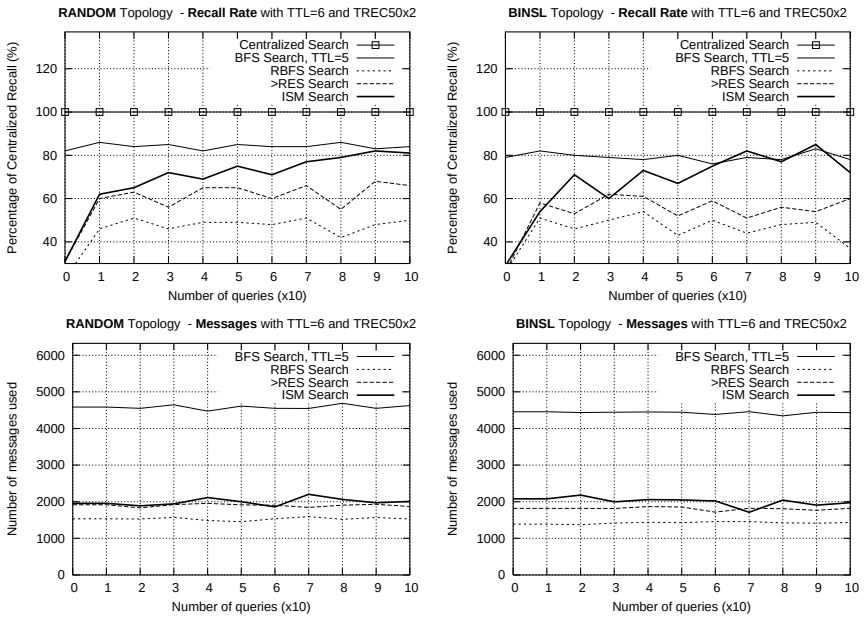

Fig. 6. a) Recall Rate (top) and **b) Messages (bottom)** for the evaluation of the TREC50x2 queryset using **Random (left)** and **BinSL (right)** topologies.

sages are able to reach 859 out of the 1000 nodes.[3] Therefore it was expected that BFS's recall rate would be less than the recall rate obtained by evaluating the whole dataset in a centralized setting. The rest techniques (i.e. RBFS, ISM and >RES), use a TTL of 6 as they offer reduced messaging, which allows us to explore the network graph deeper while maintaining low messaging. Finally, the average time to perform a query for the BFS case is in the order of 1.5 seconds but results start streaming back to the query node within the first few milliseconds. In figure 5, we plot the Aggregate Delay Δ_T for the Random (left) and BinSL (right) topology. The two figures indicate that by using BinSL any of the presented search techniques reduces the Δ_T parameter by a factor of three.

Maintaining High Recall Rates and Low Messaging: So far we have seen that by using a BinSL topology we are able to reduce the Δ_T parameter. However this single parameter is not enough in the context of information retrieval applications, as these applications are required to return the most relevant documents. Furthermore, if some search technique always explored the shortest latency neighbors then the Δ_T parameter would be minimal but the query would with very high probability get locked in some region and would not explore the

[3] With a TTL of 6 and 7, we would be able to reach 998 and 1000 nodes at a cost of $8,500$ messages/query and $10,500$ messages/query respectively.

larger part of the network graph. This would consequently reduce the recall rate which is not desirable.

In figure 6a, we plot the recall rate of the different search algorithms using the Random (left) and BinSL (right) topologies presented in the previous subsection. The figures indicate that by deploying a BinSL topology the recall rate does not significantly degrade. We note that the recall rate degrades because in the BinSL topology more query paths are short-circuited (i.e. it explores slightly smaller subgraphs of the overlay G). However this results in significant savings in the Aggregate Delay Δ_T parameter (see figure 5). In figure 6b, we plot the number of messages required by each search technique. We can see that BFS requires almost 2.5 times more messages than the other techniques. The ISM search technique on the other hand, learns from its profiling structure and guides the queries to the network segments that contain the most relevant documents. On the other hand both RBFS's and >RES's recall fluctuate, which indicates that >RES may behave as bad as RBFS if the queries don't follow some repetitive pattern.

6 Conclusions & Future Work

We considered and evaluated the impact of the use of topologically aware overlay network constructions on the accuracy and the performance of currently proposed fully distributed P2P information retrieval techniques.

Our empirical results show that the use of the topologically-aware BinSL overlay network construction technique significantly improves the latency times for all the information retrieval techniques we considered. These included both agnostic techniques (BFS, RBFS), and techniques that used past statistics (ISM, >RES), and we compared the performance of the BinSL overlay network with a random graph of the same average degree. In all cases, the accuracy remained approximately the same.

Our results clearly show the advantage of our approach. In our future work we plan to design new techniques that tightly integrate the construction of the overlay network with the actual information retrieval mechanism.

References

1. Baeza-Yates R.A. and Ribeiro-Neto B.A., "Modern Information Retrieval." ACM Press Series/Addison Wesley, New York, May 1999.
2. Batagelj V. and Mrvar A. "PAJEK - Program for large network analysis", Connections, 21:47–57, 1998.
3. Bollobás B. "Modern Graph Theory, Graduate Texts in Mathematics" vol. 184, Springer-Verlag, New York, 1998.
4. Castro M., Druschel P., Charlie Hu Y., Rowstron A. "Topology-aware routing in structured peer-to-peer overlay networks", *In IFIP/ACM Middleware'01.*
5. Chawathe Y., Ratnasamy S., Breslau L., Lanham N., Shenker S. "Making Gnutella-like P2P Systems Scalable", *In ACM SIGCOMM'03.*

6. Chu Y-H, Rao S.G., Zhang H. "A Case For End System Multicast", *In ACM SIGMETRICS'00*, Santa Clara, CA, June, 2000.
7. Clarke I., Sandberg O., Wiley B. and Hong T.W. "Freenet: A Distributed Anonymous Information Storage and Retrieval System". *Proc. of the ICSI Workshop on Design Issues in Anonymity and Unobservability*, Berkeley, CA, 2000.
8. Crespo A. and Garcia-Molina H. "Routing Indices For Peer-to-Peer Systems", *In ICDCS'02*, Vienna, Austria, 2002.
9. Cuenca-Acuna F.M. and Nguyen T.D. "Text-Based Content Search and Retrieval in ad hoc P2P Communities". Int. Workshop on Peer-to-Peer Computing, 2002
10. Garey M.R. and Johnson D.S. "Computers and Intractability: A Guide to the Theory of NP-Completeness", W.H. Freeman, 1979.
11. Gao J., and Steenkiste P. "Design and Evaluation of a Distributed Scalable Content Discovery System." *IEEE Journal on Selected Areas in Communications*, Special Issue on Recent Advances in Service Overlay Networks, 22(1):54-66.
12. Gnutella, `http://gnutella.wego.com`.
13. Hansen, T., Otero, J., McGregor, A., Braun, H-W., "Active measurement data analysis techniques", *In CIC'00*, Las Vegas, Nevada, June, 2000.
14. Iyer S., Rowstron A., Druschel P., "SQUIRREL: A decentralized, peer-to-peer web cache" *In PODC 2002*, Monterey, CA, 2002.
15. Kalogeraki V., Gunopulos D., and Zeinalipour-Yazti D. "A Local Search Mechanism for Peer-to-Peer Networks", *In ACM CIKM'02*, McLean, Virginia.
16. Lucene, The Apache Jakarta Project. http://jakarta.apache.org/lucene/
17. Lv Q., Cao P., Cohen E., Li K., and Shenker S. "Search and replication in unstructured peer-to-peer networks". ICS02, New York, USA, June 2002.
18. Napster, `http://www.napster.com/`.
19. Ratnasamy S., Handley M., Karp R., Shenker S. "Topologically-Aware Overlay Construction and Server Selection", *In IEEE INFOCOM'02*, NY, June 2002.
20. Ratnasamy S., Francis P., Handley M., Karp R., Shenker S. "A Scalable Content-Addressable Network", *In ACM SIGCOMM'01*, August 2001.
21. Rowstron A. and Druschel P., "Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems", *In IFIP/ACM Middleware'01.*
22. SETI Project Home Page, "SETI@Home", http://setiathome.ssl.berkeley.edu.
23. Stoica I., Morris R., Karger D., Kaashoek M.F., Balakrishnan H. "Chord: A scalable peer-to-peer lookup service for Internet applications", *In ACM SIGCOMM'01*, San Diego CA, August 2001.
24. Tang C., Xu Z., and Dwarkadas S. "Peer-to-Peer Information Retrieval Using Self-Organizing Semantic Overlay Networks". *ACM SIGCOMM 2003.*
25. Tsoumakos D. and Roussopoulos N. "Adaptive Probabilistic Search for Peer-to-Peer Networks". *Proc. of the Third IEEE Int. Conf. on P2P Computing*, 2003.
26. Wouhaybi R. and Campbell A. "Phenix: Supporting Resilient Low-Diameter Peer-to-Peer Topologies" *In IEEE INFOCOM'04*, Hong Kong, to appear in 2004.
27. Xu Z., Tang C., Zhang Z. "Building Topology-Aware Overlays using Global Soft-State", *In ICDCS 2003*, Providence, Rhode Island, 2003.
28. Yang B., and Garcia-Molina H. "Efficient Search in Peer-to-Peer Networks", *In ICDCS'02*, Vienna, Austria, July 2002.
29. Zeinalipour-Yazti D. and Folias T., "Quantitative Analysis of the Gnutella Network Traffic", Technical Report UC-CS-89, University of California, Riverside.
30. Zeinalipour-Yazti D., Kalogeraki V., Gunopulos D. "Exploiting Locality for Scalable Information Retrieval in Peer-to-Peer Systems", Information Systems Journal, Elsevier Publications, to appear in 2004.
31. Zhao B.Y., Duan Y., Huang L., Joseph A.D., Kubiatowicz J.D. "Brocade: landmark routing on overlay networks" In IPTPS'02, Cambridge MA, March 2002.