Semantic Small World: An Overlay Network for Peer-to-Peer Search

Mei Li Wang-Chien Lee Anand Sivasubramaniam
Pennsylvania State University, University Park, PA 16802
{meli, wlee, anand}@cse.psu.edu

Abstract

For a peer-to-peer (P2P) system holding massive amount of data, efficient semantic based search *for resources (such as data or services) is a key determinant to its scalability. This paper presents the design of an overlay network, namely* **semantic small world (SSW)***, that facilitates efficient semantic based search in P2P systems. SSW is based on three innovative ideas: 1) small world network; 2) semantic clustering; 3) dimension reduction. Peers in SSW are clustered according to the semantics of their local data and self-organized as a small world overlay network. To address the maintenance issue of high dimensional overlay networks, a dynamic dimension reduction method, called adaptive space linearization, is used to construct a one-dimensional SSW that supports operations in the high dimensional semantic space. SSW achieves a very competitive trade-off between the search latencies/traffic and maintenance overheads. Through extensive simulations, we show that SSW is much more scalable to very large network sizes and very large numbers of data objects compared to pSearch, the state-of-the-art semantic-based search technique for P2P systems. In addition, SSW is adaptive to distribution of data and locality of interest; is very resilient to failures; and has good load balancing property.*

1. Introduction

The advent of applications such as Napster and Gnutella has made peer-to-peer (P2P) systems popular for the widespread exchange of resources and voluminous information between thousands of users. In contrast to traditional client-server computing models, a node in P2P systems can act as both a server as well as a client. Despite avoiding centralized server bottlenecks and single point of failure, these decentralized systems present fundamental challenges when searching for resources (e.g., data and services) available at one or more of these numerous host nodes. Meanwhile, such decentralization mandates that these systems dynamically adapt to continuous node membership and content changes without incurring high maintenance overheads[1].

The importance of P2P searches has motivated several proposals for performing these operations efficiently. Mechanisms such as Gnutella and Random Walk [10] either flood the network or search through a single path in the network randomly. While their search costs may not be low in terms of the total number of messages and/or the number of hops traversed per search, their advantages are in the low maintenance cost, making it relatively easy to handle membership and data content changes. Improvements to better direct such messages by indexing around neighborhoods (of an overlay network), such as Local Index [16] and Neighborhood Signature [7], can enhance the performance of searches. However, membership/content changes can require additional costs to update such indexes. Further improvements to search efficiency have led to constructing overlay networks (e.g., CAN [13], CHORD [14]) that use hashed keys to direct the searches to the specific node(s) holding the requested data objects. This comes at a higher maintenance cost for updating the relevant information on membership/content changes.

While all these techniques address the scalability issue (particularly of searches) with respect to the number of nodes in the P2P system, it is equally important to address the voluminous information content of such systems. The vast repositories of information (just as in the Internet today) are simply not favorable to key-based (or keys hashed from names) searches, mandating the employment of *content/semantic-based searches*[2]. That is why search engines are popular for navigating the Internet today. The primary goal of this study is to design a P2P overlay network that supports efficient semantic based search.

To facilitate semantic based search, data (or service) objects usually are represented by a collection of attribute values which can be derived from the content or metadata of the objects. These attributes, in various formats and pre-defined domains, logically represent the semantics of the data objects. Thus, each data object can be seen as a point in a multi-dimensional *semantic space*. As a result, queries on data objects in this semantic space can be specified in terms of these attributes. There are several challenges faced by the effort to achieve our design goal. First of all, based on accumulated

1 In this paper, peer join, peer leave and peer failure are collectively referred to as membership changes.

2 We do not exploit the differences between semantic and content based searches. These two terms are used interchangeably in the paper.

knowledge of *clustered indexes* in research community, it is safe to assume that clustering data objects with similar semantics close to each other and indexing them in certain attribute order can facilitate efficient search of these data objects based on indexed attributes. Thus, a P2P overlay network designed for efficient semantic based search should be constructed in a way such that the peer hosts and data objects are organized (i.e., clustered and indexed) in accordance with the semantic space that they are located in. Secondly, for many real life applications, the number of attributes used to identify data objects and to precisely specify queries is pretty large. Thus, a well designed P2P overlay network needs to be able to facilitate efficient navigation and search in a *high dimensional space* without incurring high maintenance overhead. Finally, the P2P overlay network of our goal mandates all the good properties of a robust network such as scalability, load balance, and tolerance to peer failures.

This paper presents the design of a P2P overlay network, called *Semantic Small World (SSW)*, which overcomes the above challenges to facilitate semantic-based search[3]. This overlay network, during peer joins and leaves, dynamically clusters peers with semantically similar data closer to each other and maps these clusters in a high-dimensional semantic space into a one-dimensional *small world network* that has an attractive trade-off between search path length and maintenance costs. Further, SSW dynamically updates the overlay to take advantage of query locality and data distribution characteristics, unlike what has been proposed in the literature until now. Through extensive simulations, we demonstrate the superiority of SSW over pSearch, the state-of-the-art semantic-based search technique built on CAN in P2P systems [15].

The primary contributions of this work are three-fold: 1) We show a way to build a small world overlay network for semantic based P2P search, which is scalable to large network sizes and large numbers of data objects, while nimble enough to adapt to dynamic membership and content changes with low maintenance overheads; 2) We show a dimension reduction technique (called *adaptive space linearization* (ASL)) for constructing a one-dimensional SSW (called *SSW-1D*) to address the challenges raised by high dimensionality of semantic space; 3) We adopt an effective clustering strategy that places peer nodes based on the semantics of their data and adapts to dynamic locality of queries and user interests. As a result, SSW exhibits distinguished strength in resilience to failures and balancing the load fairly across the network even when there are hot spots.

The rest of this paper is structured as follows. Background and related work are provided in Section 2. In Section 3, the concept of SSW and technical challenges are presented. Semantic-based P2P search operation is detailed in Section 4. The simulation setup and results for performance evaluation are presented in Section 5 and Section 6, respectively. Finally, we conclude this paper and outline directions for future research in Section 7.

3 A preliminary study of Semantic Small World is reported in [9].

2. Preliminaries

In this section, we provide background on semantic space/vector and small world network, and review some studies related to our work.

2.1. Background

Semantic Space and Vector. Various digital objects, such as documents, multimedia, and genomic data can be represented and stored as data objects in P2P systems. The semantics or features of such data object can be identified by a k-element vector, namely, *Semantic Vector (SV)* (or called *feature vector* in literature). Each element in the vector represents a particular feature or attribute associated with the data object (e.g., color for an image, concept or key word for a text document) with weight representing the importance of this feature element in representing the semantics of the data object. The SV of a data object can be mapped to a point in a k-dimensional semantic space. Euclidean distance is used to represent the semantic closeness between two SVs in this paper.

Small World Network. Small world networks can be characterized by *average path length* between two nodes in the network and *cluster coefficient* defined as the probability that two neighbors of a node are neighbors themselves. A network is said to be small world if it has small average path length (i.e., similar to the average path length in random networks) and large cluster coefficient (i.e., much greater than that of random networks). Studies on a spectrum of networks with small world characteristics show that searches can be efficiently conducted when the network exhibits the following properties: 1) each node in the network knows its local neighbors, called *short range contacts*; 2) each node knows a small number of randomly chosen distant nodes, called *long range contacts*, with probability proportional to $\frac{1}{d}$ where d is the distance [5, 6]. A search can be performed in $O(log^2 N)$ steps on such networks, where N is the number of nodes in a network [6]. The constant number of contacts (implying low maintenance cost) and small average path length serve as the motivation for trying to build a small world overlay network in our approach.

2.2. Related Work

Here we review some semantic clustering techniques and focus on more details of pSearch and its dimension reduction technique, *rolling index*.

Semantic Clustering. The idea of clustering nodes with similar documents together has appeared in [1, 4, 11, 12]. Proposals in [1] and [11] rely on a centralized server or super-peers to cluster documents and nodes. Preliminary work in [4] proposes to cluster nodes with similar interest together, without discussing how to define the interest similarity amongst peers and how to form clusters. [12] relies on periodic message exchanges amongst peers to keep track of other peers with similar documents, which incurs very high message overhead. All these techniques rely on a basic assumption that data objects in a peer are highly homogeneous.

On the other hand, while taking advantage of homogeneity in data sets, SSW is suitable for both heterogeneous and homogeneous data sets.

pSearch and Rolling Index. pSearch [15] applies a dimension reduction technique, called *rolling index*, on top of CAN to realize a semantic-based search engine. Rolling index partitions the lower (but more important) dimensions into p subvectors (where each subvector consists of m dimensions, $m = 2.3 \cdot ln(N)$ and N is the total number of nodes selected to participate in the search engine) and maps the partial semantic space corresponding to each subvector into the key space of CAN. To process a semantic based search, p separate searches are performed on the CAN key space. The most similar data object(s) in the result of these p searches are returned as the answer. Rolling index can be applied on top of other overlay networks, such as CHORD, or small world network (as we demonstrate later). Although simple, rolling index incurs high index publishing overheads and search costs since each index publishing/search involves p operations with one corresponding to each subvector. In contrast, the dimension reduction technique we propose, ASL, is incorporated in the ground-up construction of SSW. It requires only a single operation for index publishing and search. In addition, ASL considers all semantic information of a search, instead of only m dimensions as in rolling index, thereby no heuristics are required to direct the search (which are required by pSearch).

3. Semantic Small World

In this section, we describe the concept of semantic small world (SSW), provide technical details on construction of SSW, and propose a solution to linearize the semantic clusters in high dimensional space into a one-dimensional SSW.

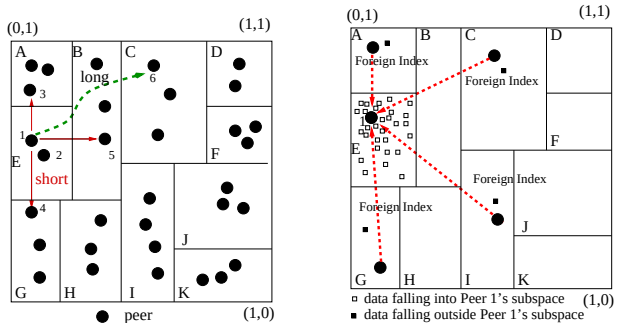
3.1. Overview

Overlay networks, the souls of P2P systems, are used to connect peer hosts into cyber communities. To facilitate efficient information search and sharing, the overlay networks of P2P systems also serve as (distributed) indexes. Since data objects captured by k-dimensional SVs can be seen as points in a k-dimensional semantic space, an idea for constructing a P2P overlay network is to organize the peer nodes and data objects in accordance with the semantic space. Thus, in addition to navigation and search, a peer node in the overlay network is responsible for management of data objects (and/or the location information of data objects stored at other peers - referred as *foreign indexes*) corresponding to a *semantic subspace*. Foreign indexes, similar to the leaf node pointers of typical tree-based index structures, provide location information regarding to where data objects are physically stored[4].

4 Due to the potential high cost of redistributing a large number of data objects within the overlay network, we choose to have a newly joined peer to publish the location information of its locally stored data objects to the other peer nodes managing the subspaces corresponding to semantics of those data objects.

Moreover, to enhance the robustness of SSW, instead of assigning each individual peer node to a separate semantic subspace, several peer nodes are actually formed as a *semantic cluster* to share the responsibility of managing a semantic subspace. These semantic clusters are self-organized into a small world network.

Corresponding to a k-dimensional semantic space, a k-dimensional SSW can be formed as follows. Each node in this k-dimensional SSW maintains s short range contacts and l long range contacts, where $s = 2k$. The short range contacts are selected to ensure that a search message issued from any node can reach any other cluster in SSW. For a k-dimensional semantic space, the short range contacts of a peer node P_1 can be intuitively set to nodes in the neighboring clusters next to P_1 in both directions of the k dimensions. The readers should note that it is possible to use an s smaller than $2k$ as long as $s \geq 2$ and the short range contacts can provide certain (encoded) ordering information to guide the navigation between any two clusters (as we will show later in our dimension reduction technique). On the other hand, the long range contacts aim at providing short cuts to reach other clusters quickly. Via short range and long range contacts, navigation in the network can be guided greedily by comparing coordinates of the destination and subspaces of the traversed nodes.



(a) Overlay structure (b) Foreign indexes

Figure 1. An illustrative example for SSW.

Figure 1 shows an example of SSW ($k = 2$). As shown in the figure, the search space is partitioned into 11 clusters after a series of peer joins and leaves. Figure 1(a) shows the overlay structure. Peer 1 in cluster E maintains short range contacts to neighboring peer clusters A, B and G and a long range contact to a distant peer cluster C. The contacts of other peers are not shown here for clarity of presentation. Figure 1(b) illustrates the concept of foreign indexes. The dark circles denote the semantic positions of peers in the semantic space. The small rectangles represent the data objects stored in Peer 1. Most of them (the white rectangles) are located in the subspace of Peer 1, but some of them (the dark rectangles) are mapped to other subspaces. Thus, location information of those data objects are stored as foreign indexes in peers of those subspaces.

There are several critical issues that need to be addressed in the design of SSW: 1) peer placement - where in the se-

mantic space should a peer node be located and what is the responsibility of a peer node? 2) cluster formation - what is the strategy for forming clusters? 3) space partition - how to partition a semantic subspace to two clusters of peer nodes? 4) dimension reduction - how to handle the maintenance issue of the overlay network if the dimensionality of corresponding semantic space is too high? The first three issues listed above are closely related to the problem of *semantic clustering*, while the last one obviously is related to the problem of *high dimensionality*. In Section 3.2, we will discuss our strategies for performing semantic clustering while introducing the tasks of constructing a k-dimensional SSW. In Section 3.3, we will describe how we linearize the k-dimensional SSW into a one-dimensional SSW in support of k-dimensional semantic space.

3.2. Construction of Semantic Small World

We now discuss how to construct a small world network depicted above. This involves three major tasks: 1) obtaining a *semantic label* that positions a peer node in the semantic space; 2) forming peer clusters in the semantic space; 3) constructing an overlay network across the logical peer clusters to form a semantic small world network.

Semantic labelling. This task is executed before or when a peer node joins the network. We assume that each node obtains the SVs of its local data objects by local computation. Then, a peer node clusters its local data objects into *data clusters* consisting of data objects with similar semantics [17]. A peer chooses the *centroid of its largest data cluster* as its semantic label (also called a *join point*) to decide which semantic subspace (and which cluster) the peer is to be placed in. While we assume a single join point here, multiple join points can be used if the peer node has sufficient resources.

Using centroid of the largest data cluster in a peer node to decide the peer's position in the semantic space has several positive effects. For example, if a node has relatively homogeneous data set (which is likely to be the case in real life), the semantic subspace where a peer resides in is also where most of its data objects fall into, thereby reducing the cost to publish foreign indexes. Moreover, the queries issued by the peers in the nearby subspace usually exhibit similar locality, i.e., a peer is likely to query for data objects with similar semantic meaning as its own data objects. Our construction of SSW exploits these characteristics naturally and still works better than other state-of-the-art P2P search techniques even without these localities (demonstrated later in the paper).

Cluster formation. In SSW, we use a preset maximum cluster size M to decide cluster boundary. When a new peer node joins the network, it navigates to and joins a cluster which accommodates its semantic label (i.e., join point). The peer nodes within a cluster know each other (directly or indirectly) by keeping track of a pre-determined number (called *out-degree*, where out-degree $\leq M$) of peers within the cluster. If the cluster size exceeds M, the cluster will be split into two based on our space partition strategy (to be discussed next) in order to maintain good clustering effect. As such, the size of semantic subspaces adapts to the density of peers and data objects in the semantic space.

Our space partition strategy is aiming at load balancing (based on data distribution within a space rather than the covered area of a space). To proceed, two peers in the cluster that are semantically farthermost from each other are selected as the seeds for the two sub-clusters. Then, peers in the cluster are alternatively assigned to the two sub-clusters based on the shortest distance to the seeds. Finally, the cluster space is partitioned at the middle point of the dimension that has the largest span between the centroids of the semantic labels of the two sub-clusters (low order dimensions are used to break ties). This is similar to how R-tree nodes are split [3]. Based on this strategy, we obtain two subspaces that have relatively equal load (in terms of the number of foreign indexes) even though the physical size of the two subspaces may not be equal. Existing overlay networks, such as CAN and CHORD, simply partition a space into two equal sized subspaces without considering the load distribution in the two subspaces.

Overlay network construction. To construct the overlay, each peer node maintains a set of short range contacts pointing to a peer in the neighboring peer clusters and a certain number of long range contacts. The long range contacts are obtained by randomly choosing a point in the semantic search space based on a distribution, $\frac{C}{d^k}$ where k is the dimensionality of the semantic space, d is the semantic distance, and C is a normalization constant that brings the total probability to 1. These extra long range contacts reduce the network diameter and transform the network into a small world with polylogarithmic search cost [8]. In addition, there are no rigid rules on which specific distant clusters should be pointed to by long distance contacts. This flexibility of long range contact selection makes SSW adapt to locality of interest easily.

3.3. Dimension Reduction

An intuitive way to support P2P applications that have complex data objects with k-dimensional SVs is to construct a k-dimensional SSW by simply assigning short range contacts in all dimensions of the corresponding semantic space. However, for a semantic space with high dimensionality (e.g., the dimensionality of semantic vector used in document retrieval is around 50-300 [2]), this approach makes maintenance costly and non-trivial due to the decentralized and highly dynamic nature of P2P systems.

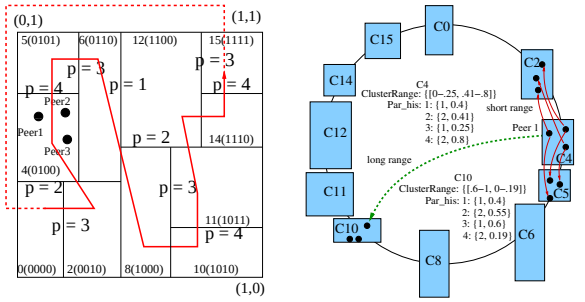
A common strategy to address the issue of high dimensionality is to perform dimension reduction. One approach, based on the idea of rolling index, is to partition the semantic vector of a data object into x disjoint subvectors where each subvector consists of y elements. Then each subvector is published to a y-dimensional SSW. Therefore, the x subvectors of a data object are mapped to x different places in this y-dimensional SSW. To process a query, the query vector is similarly partitioned into x subvectors. The query is routed to x subspaces covering each of these x subvectors and the data objects matching the query are returned as results. Al-

though simple, this approach incurs high foreign index publishing overheads and search costs.

In contrast to rolling index which reduces the dimensionality of semantic vectors, our approach is to reduce the dimensionality of the overlay network. We observed that while a high dimensional overlay network is search efficient (due to the large number of possible routes among peer nodes), the maintenance complexity and overhead of such a network is also overwhelming. Thus, we construct an overlay network of low dimensionality to support the connectivity of peer nodes and the function of semantic based search. This idea is realized by *adaptive space linearization* (ASL), that linearizes the clusters in high dimensional space into a one-dimensional SSW (termed as SSW-1D) during the process of cluster split in SSW construction. ASL preserves the semantic proximity among clusters as much as possible. Note that ASL serves similar goals as the well known space filling curves such as Hilbert curve, Z-curve, etc. However, these existing space filling curves can only be employed to map a regularly coordinated high-dimensional space to a low-dimensional space. In our case, the high-dimensional semantic space is adaptively (irregularly) partitioned according to data density. Therefore, these space filling curves can not be employed naturally to reduce dimension of SSW.

SSW-1D is constructed as a double linked list consisting of semantic clusters connected via two short range contacts of each peer node. In addition to this linear network structure that provides basic connectivity, long range contacts provide short cuts to other clusters which facilitate efficient search. While the original semantic space has been partitioned and then linearized, the clusters in SSW-1D are still corresponding to their original semantic subspace of high dimensionality. Thus, a crucial requirement for SSW-1D is how to encode a naming space to facilitate efficient navigation based on their high dimensional semantic information (i.e., SVs) corresponding to the original semantic space. In other words, a search needs to be able to reach its destination in semantic space (a cluster) quickly. This issue is addressed as follows.

To maintain the 1-1 mapping between the naming of clusters in SSW-1D and their semantic subspaces, we use a binary bit string (called *cluster ID*) to name the cluster. Each peer maintains a variable, *Par_Bit*, which initially points to the most significant bit of the cluster ID. Par_Bit indicates the bit to be set (to 0/1) in the next cluster split. After each split, the two sub-clusters decrease their Par_Bit by one (reset Par_Bit to the next less significant bit). The first peer cluster in the network sets all bits of its cluster ID to 0. When peers continue to join the network and eventually trigger a cluster splitting, the two sub-clusters obtain IDs by setting the bit pointed by Par_Bit separately and retaining all other bits the same as the ID of the original cluster. The sub-cluster that has smaller centroid along the partition dimension obtains an ID with the bit pointed by Par_Bit set to 0 and the other one obtains an ID with the bit pointed by Par_Bit set to 1. The same process is employed as more peers join the system and invoke more splits. The cluster merging process can be done reversely, so we do not go into details.



(a) ClusterID (b) SSW-1D

Figure 2. An illustrative example of SSW-1D.

Figure 2(a) shows a snapshot of the system where the whole semantic space is partitioned into 11 clusters with the cluster IDs indicated in the figure. We illustrate the process in a 2-dimensional space where the vertical lines represent the first dimension and the horizontal lines represent the second dimension. We assume that the name space is 4-bit long. In this example, the semantic space is first partitioned along the vertical line denoted by ”p = 1”. At this point, peers at the left side and right side of this line obtain ID ”0000” and ”1000”, respectively. Then the left side is partitioned along the horizontal line as indicated by ”p = 2”. At this point, peers at the lower left side and top left side obtain ID ”0000” and ”0100”, respectively. The solid line shows the order of the assigned cluster IDs, while the dashed line (naturally created since SSW-1D is a double linked list) indicates that a search can be performed bi-directionally. Figure 2(b) illustrates SSW-1D built upon the naming scheme described above. A peer in cluster 4 maintains short range contacts to neighboring peer clusters 2 and 5. It also maintains a long range contact to a distant peer cluster 10. The contacts of other peers are not shown here for clarity of presentation.

4. Peer-to-Peer Search

This section details the search operation in SSW-1D[5]. Table 1 summarizes the information maintained in each peer node. ClusterState consists of ClusterRange, specifying the semantic subspace covered by the cluster that this peer node resides in, ClusterSize, indicating the current size of its cluster, Par_His, recording the previous partitions that this peer has been involved in, and Par_Bit, indicating the position of next bit to be set for future sub-clusters during next partition. Par_His consists of tuples of ⟨Dimension, Par_Pt⟩ which indicates the partition point along the specified dimension. NeighborList, for intra-cluster search, stores *out-degree* NodeIds of peer nodes within the same cluster. ShortContact and LongContact are self-explanatory. Each contact consists of a NodeId and the ClusterRange of the subspace that

5 Due to space constraint, readers are referred to [8] for the detail of maintenance.

the pointed node resides in. ForeignIndex, for the location information of data objects stored at other nodes, consists of a set of semantic vectors of data objects as well as the NodeIDs of their source nodes.

ClusterState: {ClusterRange, ClusterSize, Par_His, Par_Bit}
NeighborList: {NodeId}
ShortContact: {NodeId, ClusterRange}
LongContact: {NodeId, ClusterRange}
ForeignIndex: {Semantic Vector, NodeId}

Table 1. Data structure maintained at each peer

To initiate a semantic-based search, a *search semantic vector* (denoted as Q) is generated based on the query. The search process consists of two stages: *flooding search* and *navigation*. Correspondingly, the search operation at a peer node has two modes: *search-within-cluster* and *search-across-cluster*. When a message is received, a peer node will first check whether Q falls within the range of its cluster. If that is the case, it starts search-within-cluster mode by flooding the message to peers in its NeighborList (except for the one from whom the message was received[6]). Then the data object with highest similarity to the query is returned as the result. Search-across-cluster mode is invoked when Q is not within the range of current peer's cluster. In this case, a pseudo-cluster-name (PCN), the estimated ID for the cluster covering the search semantic vector, is first calculated for Q based on the partition history (Par_His) stored at this peer (as explained later). We call the process to calculate PCN for Q as *PCN estimation*. The search-across-cluster mode is continued by forwarding the search message to the contact with the shortest naming distance to the PCN. The above process is repeated until the cluster whose semantic subspace covering Q is reached.

The algorithm for PCN estimation is illustrated in Algorithm 1. We first set all the bits of PCN to 0. Iterating through the Par_His of current peer (Peer i in the algorithm), the bits of PCN (starting from the most significant bit B) are set as the same value of corresponding bits of Peer i's Cluster ID, C_i, as long as Q confirms to the same Par_His entry (see Lines 3-4 in Algorithm 1). Otherwise, the corresponding bit is set to a different value and the PCN estimation process at Peer i stops (see Lines 6-7) since this peer does not have further details about the PCN.

Here, we show an example to illustrate the search process in SSW-1D. Let's go back to Figure 2(b). Assuming that Peer 1 in Cluster 4 wants to search for data objects based on Q [0.9,0.3], it first checks its own cluster range. Since [0.9,0.3] is not within the subspace of the cluster, Peer 1 then estimates the PCN for the query as 8 using above algorithm. Peer 1 checks its contacts and forwards the search to the closest peer node, which is a peer in Cluster 10 in this example. When the query reaches a peer in Cluster 10, this peer re-estimates the PCN for the query since the query is still not in its cluster range. This time, the query PCN is estimated as 11. The search is finally forwarded to a peer in Cluster 11, which finds Q within its own cluster range, so it floods the cluster for search results.

6 A sequence number is attached to each search message so that a node can recognize and drop a search message that appeared before.

Algorithm 1 PCN estimation.

PCN estimation for Q at Peer *i* (B is the size of bit strings used for Cluster ID)

1: **for** $x = B$ to $Par\_Bit + 1$ **do**
2: Obtain partition dimension d and partition point p from Peer i's Par_His$_x$.
3: **if** $i.ClusterRange_d \leq p$ and $Q_d \leq p$ or $i.ClusterRange_d > p$ and $Q_d > p$ **then**
4: $PCN_x = Ci_x$.
5: **else**
6: $PCN_x = 1 - Ci_x$.
7: Break.
8: **end if**
9: **end for**

Usually, the PCN resolves very fast as the message moves towards the destination, but occasionally, it takes more than one step to resolve a bit in PCN. We group the step(s) to resolve one bit of PCN as a *PCN resolving phase*. A search may need to go through multiple PCN resolving phases, where each phase brings the search message half-way closer to the target. The following theorem obtains the search path length for SSW-1D.

Theorem 1 *For a k dimensional space, given a SSW-1D of N nodes, with maximum cluster size M and number of long range contacts l, the average search path length for search across clusters is $O(\frac{log^2(2N/M)}{l})$.*

Proof: Omitted due to space constraint. Interested readers please refer to [8].

Note that even though dimension reduction has been used to build a tractable SSW-1D overlay, search is conducted using all dimensions of the search semantic vector.

During the search process, SSW adapts to locality of users interests by maintaining a search-hit list at each peer node that consists of the nodes which have search hits in the past X searches issued by this peer. For every X searches, a node replaces the long range contact having the lowest hit rate with the entry in the search-hit list having the highest hit rate with probability of $do/(do + dn)$ where do and dn represent the naming distance of the old long range contact and the candidate long range contact to current peer, respectively.

5. Performance Evaluation

We move on to evaluate SSW's benefits using extensive simulations. We compare SSW-1D (we refer it as SSW in this section for simplicity) with pSearch, the state-of-the-art in semantic-based P2P search. The goal of a search is to find a data object semantically similar to the specified query. Based on [15], pSearch takes 4 groups of the most important dimensions, each with m dimensions (i.e., $p = 4$, $m = 2.3lnN$). In addition, we also implement the rolling index used in pSearch

on top of m-dimensional small world network, called as *small world rolling index (SWRI)*, to compare the effectiveness of our proposed dimension reduction method, ASL, vs. rolling index in terms of search performance, maintenance cost and result quality. For fair comparison, the long range contact update (as described in Section 4) is turned off if unspecified otherwise. The simulation setup, parameters and performance metrics are explained below.

5.1. Simulation Setup

The simulation is initialized by having one node pre-exist in the network and then injecting node join operations into the network till the network reaches a certain size (N). After this point, a mixture of operations including peer join, peer leave and search are randomly (based on certain ratios) injected into the network. This is also when statistics collection begins. On the average, each peer issues 100 searches during each run of the simulation. The proportion of join to leave operations is kept the same to maintain the network at approximately the same size. The simulation parameters, their values and the defaults (unless otherwise stated) are given in Table 2. Most of these parameters are self-explanatory. More details for some of the parameters are given below.

	Descriptions	**Values, default**
N	Number of nodes in the network	256 - 16K, $\underline{1K}$
l	Number of long range contacts	$\underline{4}$
M	Size of peer clusters	1 - 1024, $\underline{8}$
x	Out-degree within peer clusters	$\underline{4}$
n	Number of data records per peer	1 - 100, $\underline{100}$
α_{d1}	Skewness of Dataseed-Zipf	0 - 1.0, $\underline{0}$
α_{d2}	Skewness of Data-Zipf	0 - 1.0, $\underline{0}$
γ	Percentage of join/leave operations	0% - 50%, $\underline{20\%}$
α_{q1}	Skewness of Queryseed-Zipf	0 - 1.0, $\underline{0}$
α_{q2}	Skewness of Query-Zipf	0 - 1.0, $\underline{0}$

Table 2. Parameters used in the simulations

Data Parameters: Without loss of generality, we set the dimensionality of SV (and the semantic space) to 100. The data set is defined by the number of data objects per peer (n) and the data distribution in the semantic space, which is determined by two factors: 1) semantic distribution of data between the different peers; and 2) semantic distribution of data objects at a single peer. The former controls the data hot spots in the system and the latter controls the semantic similarity between data objects at a single peer, namely *semantic closeness*. To model both factors, we associate a Zipf-distribution each, *Dataseed-Zipf* for the former and *Data-Zipf* for the latter, since this distribution provides a ready parameter (α_{d1} and α_{d2} respectively) for controlling the skewness/uniformity (the skewness is high when these values are 1, and the distribution becomes uniform when these are 0). We first draw a seed for each peer node following the Zipf distribution controlled by α_{d1}. This serves as the centroid around which the actual data objects for that peer are composed following α_{d2}.
Workload Parameters: The percentage of join/leave operations (note that the ratio of joins:leaves is itself set at 1:1) is a parameter (γ) that we control. In addition, similar to data parameters, we consider two factors in generating queries: distribution of queries emanating across the nodes in the system and the skewness of the queries emanating from a single peer. The former controls query hot spots (i.e. more users are interested in a few data items) in the system and the latter controls the locality of interest for a single peer, namely *query locality* (i.e. a user is more interested in one part of the semantic space). As with the data distributions, we use two Zipf distributions with parameters α_{q1} (for *Queryseed-Zipf*) and α_{q2} (for *Query-Zipf*) to control the skewness, i.e., α_{q2} captures the skewness of the queries around the centroid that is generated by α_{q1}.

5.2. Metrics

While the main focus of this paper is to improve search performance at a minimum overhead, we also try to explore the strengths and weaknesses of SSW in other aspects, such as fault tolerance and load balance. We use the following metrics for our evaluations:
Search path length is the average number of logical hops traversed by search messages to the destination.
Search cost is the average number of messages incurred per search. Flooding techniques like Gnutella may have short path length, but their search cost is high.
Maintenance cost is the number of messages incurred per membership change, consisting of **overlay maintenance cost** and **foreign index publishing cost**. Since the size of different messages (join, query, index publishing, cluster split, cluster merge) is more or less the same (dominated by the size of one SV (400 bytes)), we focus on the number of messages in the paper.
Search failure ratio is the percentage of unsuccessful searches that fail to locate existing data objects in the system.
Index load is the number of foreign index entries maintained at a node.
Routing load is the number of search messages that a node processed.
Result quality is to measure the quality of the returned data object. To calculate this metric, we first calculate the normalized dissimilarity (Euclidean distance)[7], *dissim_real*, between the query and the result returned by SSW (or pSearch/SWRI), and the normalized dissimilarity, *dissim_ideal*, between the query and the data object that is most similar to the query in the system. Then we use $1 - (dissim\_real - dissim\_ideal)$ to represent result quality. When the difference between dissim_ideal and dissim_real is very small, the result quality is high. For pSearch/SWRI, the most similar data object is returned from each of the p partial semantic spaces and the most similar one among those p data objects is returned as the result and used for calculating the result quality.

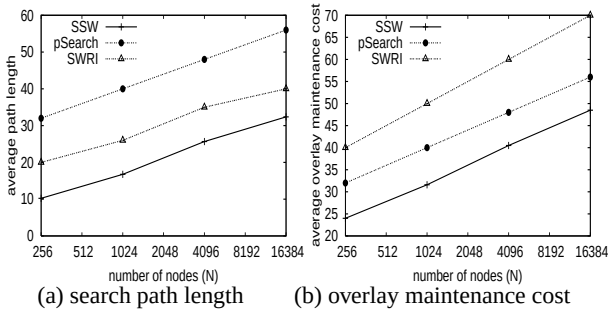
7 To perform the normalization, we divide the Euclidean distance between two vectors by $\sqrt{k}$, which is the maximum Euclidean distance between two vectors in the semantic space.

6. Simulation Results

In this section, we first demonstrate the scalability of SSW in terms of the size of the network and the number of data objects in the system. This is followed by an examination of the effect of cluster sizes on SSW. The benefits of constructing overlay based on semantics and updating long range contacts is subsequently illustrated with different workload behaviors. Lastly, we show the strength of SSW in tolerating peer failures and balancing the load.

6.1. Scalability

In terms of scalability to network size, we vary the number of nodes from 2^8 to 2^{14} to evaluate the search efficiency and maintenance cost of SSW. In our preliminary study [9], we find that SSW with 4 long range contacts has reasonable trade-off between search efficiency and maintenance overhead for most of the γ settings, and we use this value in the experiments. Since pSearch does not use any clustering, we disable the clustering feature of SSW (i.e., cluster size is set to 1) in these experiments. A later experiment will evaluate SSW with various cluster sizes and show that it can perform even better with appropriate cluster sizes.



(a) search path length (b) overlay maintenance cost

Figure 3. Comparing the network size scalability of the schemes with respect to search path length and maintenance costs.

Figure 3(a) shows the average path length. Since the size of peer clusters is set to 1 in this experiment, there is no flooding within a cluster and the average search path length for SSW represents the search cost as well. The search path length for SSW increases slowly with the size of network, confirming search path length bound in Theorem 1. In addition, the constant hidden in the big-O notation is much smaller than 1 as shown in the figure. The slope of pSearch's path length is close to SSW with 4 long range contacts but with a much higher offset. In fact, the search path length of SSW is about 40% shorter compared to pSearch at network size 16K. The search path length of SWRI is between pSearch and SSW. This confirms the shorter path length of small world network compared to CAN. In addition, it confirms the effectiveness of ASL vs. rolling index in terms of search path length.

Overlay maintenance cost is proportional to the number of states maintained at each peer, which are 20, 24 (20 short range contacts and 4 long range contacts), 6 (2 short range contacts and 4 long range contacts) for pSearch, SWRI and SSW respectively. Figure 3(b) shows the overlay maintenance cost for the same experiments as Figure 3(a). These two figures confirm our expectation that compared to pSearch, SSW can achieve better search performance with much smaller number of states maintained per peer.

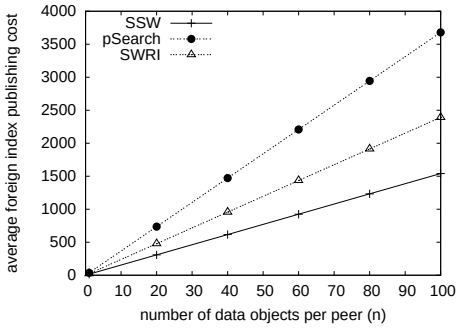
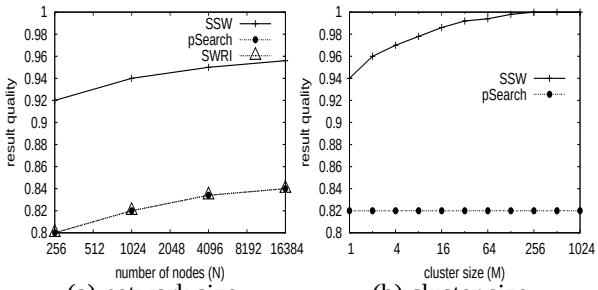


Figure 4. Comparing the foreign index publishing costs as a function of the number of data objects per peer.

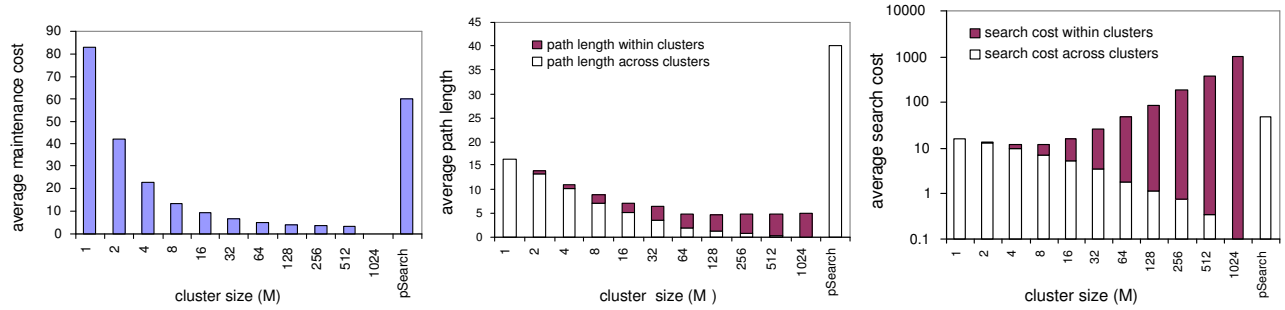
The other maintenance cost to consider is the overhead of publishing foreign index at peer joins (apart from the cost shown in Figure 3(b)). This cost is proportional to the number of data objects that need to be published, and the corresponding relationship is shown in Figure 4. Due to the fact that pSearch/SWRI have to publish a data object multiple times, the index publishing costs for pSearch/SWRI are much higher than SSW. In addition, the index publishing cost for SWRI is lower than pSearch due to the fact that small world network has shorter search path length compared to CAN as demonstrated earlier in Figure 3(a). Note that these results for SSW are conservative since $\alpha_{d2} = 0$ (uniform data distribution) and the overheads are likely to be lower with any skewness (as will be shown later).



(a) network size (b) cluster size

Figure 5. Comparing result quality of the schemes.

Figure 5(a) shows that, as expected, the result quality of SSW is higher than pSearch and SWRI since all semantic dimensions are considered in ASL while rolling index only considers a portion of the 100 semantic dimensions in each of the p partial semantic spaces. Even though pSearch/SWRI



(a) maintenance cost (b) path length (c) search cost

Figure 6. Studying the effect of cluster size (M) for SSW.

choose the best answer from the p objects returned from the p searches, result quality of all those p objects might not be as good as the one returned by SSW. This confirms the benefits of ASL vs. rolling index. pSearch and SWRI actually have the same result quality since both apply rolling index with the same setting (i.e., same m and p).

This set of experiments confirms the scalability of SSW. It also confirms our expectations that ASL is a better dimension reduction method than rolling index in terms of various aspects, such as search cost, index publishing cost, and result quality. In the remaining experiments, we only compare with pSearch for presentation clarity.

6.2. Clustering Effects

Until now the size of the peer cluster (M) has been set at 1. When M is larger, cluster splits or merges occur less frequently, resulting in lower overlay maintenance costs. Further, the total number of clusters in the system decreases with larger cluster sizes, thereby reducing searches across clusters. The down-side of large sized clusters is the higher search cost within a cluster (due to flooding).

The effect of the cluster size on the maintenance cost, overlay navigation path length/cost (across clusters), and flooding search path length/cost (within clusters) are given in Figure 6. The cluster size is varied between 1 and 1024 (the whole network is one big cluster). In these graphs, the bars for pSearch (the last bar) is also given for easier comparison. As expected, the maintenance cost decreases when the cluster size increases (drops by 75% when the size increases from 1 to 4). The path length, though decreases slightly (because of the steeper drop in path length across clusters), is not as sensitive to the cluster size compared to the overall search cost (note that the third graph has y-axis in log scale). This is because the effect of the flooding within the cluster dominates for larger clusters. Within a spectrum of cluster sizes between 2 and 16, SSW does better than the size of 1 (whose results were presented in the previous section) in terms of all maintenance cost, path length and search cost.

With larger cluster size, the semantic subspace to be examined for each query increases. Therefore, the quality of result is likely to increase with the cluster sizes. This is confirmed by Figure 5(b).

We have also conducted simulations by considering different mixes of the join/leave and search operations. Based on the results (omitted due to space constraint), we set the cluster size to 8 for the rest of the simulations.

6.3. Semantic Closeness

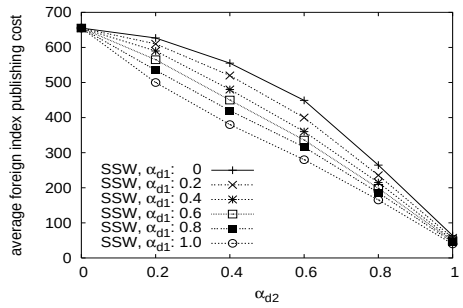
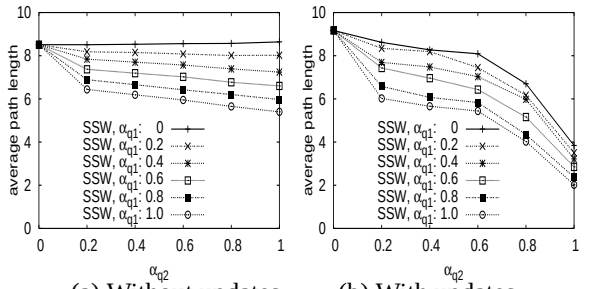


Figure 7. Effect of data distributions. Cost for pSearch is much higher (not shown for clarity).

In SSW, a peer selects the semantic centroid of its largest local data cluster as the join point when it joins the network. The rationale is that when data is more skewed around the centroid, fewer foreign indexes for data objects need to be published outside the cluster, thereby reducing overheads. To better understand the impact of semantic closeness on the foreign index publishing cost, we synthesize various data distributions at a peer by varying, α_{d2}, the skewness for Data-Zipf from 0 to 1. In addition, we also vary, α_{d1}, the skewness for data seed distribution (Dataseed-Zipf) from 0 to 1 to observe the effect of data hot spots. The effect of α_{d2} on the foreign index publishing costs for SSW are shown in Figure 7 with different values of α_{d1}. As pointed out, a higher skewness lowers the foreign index publishing cost of SSW significantly. pSearch's foreign index publishing costs are in the range of 3500 and only decrease slightly under skewed data distribution. We omit the plot for pSearch from this figure to avoid distorting the graph.



(a) Without updates (b) With updates

Figure 8. Effect of query characteristics. Cost for pSearch is much higher (not shown for clarity).

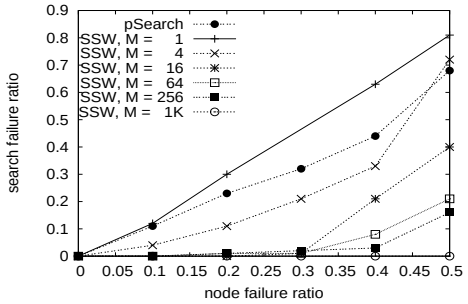


Figure 9. Effect of node failure ratio on the failure of search operations ($\gamma = 0\%$).

In SSW, long range contacts can be updated based on query history to exploit query locality. To study this improvement, we synthesize different query workloads by varying, α_{q2}, the skewness for Query-Zipf from 0 to 1. We also vary, α_{q1}, the skewness for Queryseed-Zipf to observe the effect of query hot spots. Figure 8 compares the search path length of SSW (a) without updates and (b) with updates of long range contacts for every 10 searches (based on what described in Section 4). Without any updates, query locality has little impacts on the results. With long range contact updates, however, query locality significantly enhances search performance. For instance, we see nearly a 78% reduction in path length when α_{q2} increases from 0 to 1.0 with α_{q1} set to 1.0. pSearch's result (plot is not shown here) is similar to the one without update except that pSearch's path length is much higher (in the range of 40).

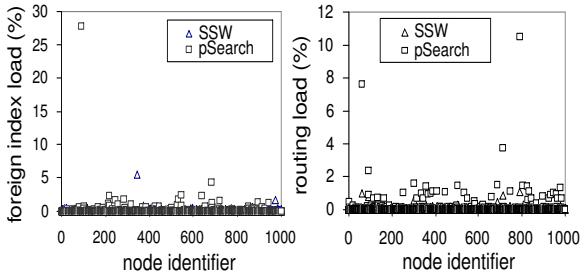
6.4. Tolerance to Peer Failures

Peer failure is a common event in P2P systems. Thus, a robust system needs to be resilient to these failures. To evaluate the tolerance of SSW to peer failures, a specified percentage of nodes are made to fail after the network is built up. We then measure the ratio of searches that fail to find data objects existing in the network (we do not consider failures due to the data residing on the failed nodes).

Figure 9 shows the fraction of searches that fail as a function of the number of induced peer failures (from 0% to 50%). Since the fault tolerance of SSW is largely dependent on the cluster size, we also consider different values for M (the cluster size) in these experiments. Even though each peer in pSearch maintains a large number of states (20), the search failure ratio grows rapidly with the number of node failures. At cluster size of 1, SSW with much smaller number of states (2 short range contacts and 4 long range contacts) maintained per peer has similar search failure rate as pSearch. However, moving to a cluster size of 4 substantially improves SSW's fault tolerance. Beyond sizes of 4, the search failure ratio, even with as high as 30% node failure, is very close to 0. These results reiterate the benefits of forming clusters.

6.5. Load Balance

We evaluate the load balance of SSW from two aspects: index load and routing load. For the index load, we evaluate the distribution of the foreign index maintained at each peer under two extreme data distribution patterns: uniform (with both α_{d1} and α_{d2} set to 0) and skewed (with both α_{d1} and α_{d2} set to 1). Since the load is evenly balanced under the uniform distribution for both pSearch and SSW, we present only the index load for the skewed distribution in Figure 10(a). As we expected, pSearch has a much more uneven index load distribution compared to SSW (more rectangles with a higher load than triangles). In fact, Peer 87 in charge of a hot data region in pSearch stores about 28% of the index load of the whole system. In contrast, SSW displays a relatively even distribution of index load even under this skewed data set, confirming our intuition that placing peers in the semantic space in accordance with their local data objects can effectively partition the search space according to data density.



(a)index load (b)routing load

Figure 10. Distribution of foreign index load and routing load amongst the nodes. More points with heavy load are indication of imbalance.

Similarly, we have varied the query distribution in order to study the routing load distribution across the nodes,

again with uniform and skewed distributions (this time with queries). We only present the results for the skewed queries (α_{q1} and α_{q2} set to 1) in Figure 10(b). We find that the routing load is more evenly distributed in SSW compared to pSearch. This is due to the randomness of the long range contacts and the good balance within a cluster itself.

7. Conclusion

In this paper, we propose a new P2P overlay network, semantic small world (SSW), to facilitate efficient semantic based search. SSW is unique in the aspect that the overlay network is constructed based on a semantic space. Peer nodes are clustered and organized in accordance with the semantics of data objects stored locally. These peer clusters then self-organize into a small world network which has efficient search performance with low maintenance overhead.

For many real life applications, the number of attributes used to identify data objects and to precisely specify queries is pretty large. The high dimensionality of semantic space represents a primary challenge for maintaining an overlay that facilitates efficient traverse and search in such a space. In this paper, we proposed a dynamic dimension reduction method, called adaptive space linearization (ASL), to construct a one-dimensional SSW that operates in a high dimensional semantic space. ASL, based on the idea of reducing dimensionality of the overlay network, has been shown to be more effective than rolling index, a technique that reduces the dimensionality of the semantic vectors.

SSW has many highly desirable features. It facilitates efficient search without incurring high maintenance overhead. By placing and clustering peers in the semantic space based on the semantics of their data objects, SSW adapts to distribution of data automatically, gains high resilience to peer failure and balances index and routing load nicely. In addition, SSW harnesses the locality of queries and user interest naturally. All of the above advantages of SSW are verified through extensive simulation. We believe that SSW can have a significant impact on the deployment of large scale P2P applications.

We are tuning the performance of SSW further and exploiting resource heterogeneity amongst peers by dynamically adjusting number of join points, long range contacts and cluster size. Finally, we are investigating how to extend SSW for data management in ad hoc and sensor networks.

References

[1] M. Bawa, G. S. Manku, and P. Raghavan. SETS: Search enhanced by topic segmentation. In *Proceedings of ACM SIGIR*, pages 306–313, July 2003.

[2] M. W. Berry, Z. Drmac, and E. R. Jessup. Matrices, vector spaces, and information retrieval. *Society for Industrial and Applied Mathematics Review*, 41(2):335–362, 1999.

[3] A. Guttman. R-trees: A dynamic index structure for spatial searching. In *Proceedings of ACM SIGMOD*, pages 47–54, 1984.

[4] A. Iamnitchi, M. Ripeanu, and I. T. Foster. Locating data in (small-world?) peer-to-peer scientific collaborations. In *Proceedings of International Workshop on Peer-to-Peer Systems (IPTPS)*, pages 232–241, March 2002.

[5] J. Kleinberg. Navigation in a small world. *Nature*, 406(845), August 2000.

[6] J. Kleinberg. The small-world phenomenon: an algorithm perspective. In *Proceedings of ACM Symposium on Theory of Computing*, pages 163–170, May 2000.

[7] M. Li, W.-C. Lee, and A. Sivasubramaniam. Neighborhood signatures for searching P2P networks. In *Proceedings of International Database Engineering and Application Symposium (IDEAS)*, pages 149–158, July 2003.

[8] M. Li, W.-C. Lee, and A. Sivasubramaniam. Semantic Small World: An overlay network for peer-to-peer search. Technical Report CSE 04-016, Pennsylvania State University, 2004.

[9] M. Li, W.-C. Lee, A. Sivasubramaniam, and D. L. Lee. A small world overlay network for semantic based search in P2P systems. In *Proceedings of Workshop on Semantics in Peer-to-Peer and Grid Computing (SemPGrid), in conjunction with the World Wide Web Conference (WWW)*, May 2004.

[10] Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker. Search and replication in unstructured peer-to-peer networks. In *Proceedings of ACM International Conference on Supercomputing*, pages 84–95, June 2002.

[11] W. Nejdl, M. Wolpers, W. Siberski, C. Schmitz, M. T. Schlosser, I. Brunkhorst, and A. Lser. Super-peer-based routing and clustering strategies for RDF-based peer-to-peer networks. In *Proceedings of International World Wide Web Conference (WWW)*, pages 536–543, May 2003.

[12] C. H. Ng, K. C. Sia, and C. H. Chang. Advanced peer clustering and firework query model in the peer-to-peer network. In *Proceedings of International World Wide Web Conference (WWW)*, May 2003, Poster.

[13] S. Ratnasamy, P. Francis, M. Handley, R. M. Karp, and S. Schenker. A scalable content-addressable network. In *Proceedings of ACM SIGCOMM*, pages 161–172, August 2001.

[14] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for Internet applications. In *Proceedings of ACM SIGCOMM*, pages 149–160, August 2001.

[15] C. Tang, Z. Xu, and S. Dwarkadas. Peer-to-peer information retrieval using self-organizing semantic overlay networks. In *Proceedings of ACM SIGCOMM*, pages 175–186, August 2003.

[16] B. Yang and H. Garcia-Molina. Improving search in peer-to-peer networks. In *Proceedings of International Conference on Distributed Computing Systems (ICDCS)*, pages 5–14, July 2002.

[17] T. Zhang, R. Ramakrishnan, and M. Livny. Birch: An efficient data clustering method for very large databases. In *Proceedings of ACM SIGMOD*, pages 103–114, June 1996.