Supporting Ad-Hoc Resource Sharing on the Web: A Peer-to-Peer Approach to Hypermedia Link Services

JING ZHOU
Communication University of China
WENDY HALL and DAVID C. DE ROURE
University of Southampton
and
VIJAY K. DIALANI
IBM Almaden Research Center

The key element to support ad-hoc resource sharing on the Web is to discover resources of interest. The hypermedia paradigm provides a way of overlaying a set of resources with additional information in the form of links to help people find other resources. However, existing hypermedia approaches primarily develop mechanisms to enable resource sharing in a fairly static, centralized way. Recent developments in distributed computing, on the other hand, introduced peer-to-peer (P2P) computing that is notable for employing distributed resources to perform a critical function in a more dynamic and ad-hoc scenario. We investigate the feasibility and potential benefits of bringing together the P2P paradigm with the concept of hypermedia link services to implement ad-hoc resource sharing on the Web. This is accomplished by utilizing a web-based Distributed Dynamic Link Service (DDLS) as a testbed and addressing the issues arising from the design, implementation, and enhancement of the service. Our experimental result reveals the behavior and performance of the semantics-based resource discovery in DDLS and demonstrates that the proposed enhancing technique for DDLS, topology reorganization, is appropriate and efficient.

Categories and Subject Descriptors: H.3.4 [**Information Storage and Retrieval**]: Systems and Software—*Distributed systems*; H.5.4 [**Information Interfaces and Presentation**]: Hypertext/Hypermedia—*Architectures*

This research was conducted while J. Zhou and V. K. Dialani were affiliated with the School of Electronics and Computer Science, University of Southampton, UK.
Authors' addresses: J. Zhou, School of Computer and Software, Communication University of China, Beijing 100024, China; email: zhoujing@cuc.edu.cn; W. Hall, D. C. De Roure, School of Electronics and Computer Science, University of Southampton, Southampton, SO17 1BJ, UK; email: {wh,dder}@ecs.soton.ac.uk; V. K. Dialani, IBM Almaden Research Center, San Jose, CA 95120; email: vdialani@us.ibm.com.

General Terms: Algorithms, Design, Performance

Additional Key Words and Phrases: Distributed dynamic link service, open hypermedia, peer-to-peer (P2P), semantic search, reorganization

ACM Reference Format:

Zhou, J., Hall, W., De Roure, D. C., and Dialani, V. K. 2007. Supporting ad-hoc resource sharing on the Web: A peer-to-peer approach to hypermedia link services. ACM Trans. Intern. Tech. 7, 2, Article 11 (May 2007), 29 pages. DOI = 10.1145/1239971.1239975 http://doi.acm.org/10.1145/1239971.1239975

1. INTRODUCTION

As sharing of resources between a community of Web users [Berners-Lee 1996] cannot always be arranged in advance, the need for supporting resource sharing in a dynamic ad-hoc context is demanding. We believe the key issue to realize such a vision is resource discovery. As the most widely used and successful hypermedia system [Lowe and Hall 1999], the Web has no full-text search facilities of its own and relies on external search mechanisms. Open hypermedia systems (OHSs) [Wiil 1997], among others, have the potential of supporting resource discovery on the Web. Hypermedia aims to facilitate access to and manipulation of information by using various relationships which can be instantiated as hypermedia links, or links, between elements of information. Every OHS requires a link service to maintain and manipulate links separately (in linkbases[1]) from the documents they describe. Link services implement resource sharing by sharing links among involved parties and carry out link discovery by resorting to a central service directory (see Section 2.1). They are mainly designed to operate in a fairly static environment in which some form of centralized control is present and hence are incapable of directly supporting resource sharing on the Web on an ad hoc basis.

Recent advances in distributed computing introduced peer-to-peer (P2P) computing[2] [Clark 2001]. The novelty of P2P technologies developed over the last few years is that they allow Internet-connected personal computers to play more important roles than those played by client-server or master-slave systems. P2P computing attracts considerable attention from research communities due to its potential for supporting activities among groups of people—it provides individual nodes with autonomy and control over their own resources and empowers sharing of resources in a decentralized, scalable, and ad-hoc fashion. For instance, P2P file-sharing systems can achieve a potentially unlimited area for the exchange and sharing of files and music clips by aggregating storage from distributed participants and directly transferring resources of interest between the provider and the requestor.

In this article, we bring together P2P with hypermedia link services. The vision is to support a virtual research community in which people with similar knowledge backgrounds maintain network accessible documents for sharing.

[1]Linkbases refer to the link databases that contain all the information about link availability.

[2]Although the term P2P computing is new, the basic P2P technology dates back to at least 1979 when USENET was originally implemented.

In this application scenario, people are allowed to analyze, categorize, and annotate documents. The annotation information serves to discover documents of interest. Because people may have different viewpoints on the same document, sharing information about documents enables them to understand other peers' opinions on the same concept that a document conveys by means of the way the document is annotated. A distinctive feature of such a scenario is ad-hoc: resources available at any particular time is unknown and unpredictable and both services and resources from a particular user are more probabilistic than deterministic. To make this a reality, we further identified the following key issues: routing queries to locate resources, organizing and manipulating resources to facilitate discovery and presentation, and enhancing resource discovery performance.

We describe the development of a Web-based Distributed Dynamic Link Service (DDLS). The DDLS has its roots in the DLS (Distributed Link Service) philosophy (see Section 2.3.1) and adopts an unstructured P2P architecture (see Section 2.2.2) to allow users to possess their own linkbases and share the linkbases with distributed online peers. In contrast to the DLS, the DDLS is applicable to an environment lacking a centralized control for resource publishing, discovery, and sharing and is characterized by intermittent availability of resources. We represent the resources of each peer in the DDLS in terms of a topic vector and apply a distance-based semantic search algorithm to resource discovery (see Section 3.1.4). Further, we propose the use of the exponential decay function and the naive estimator, both of which rely on the local knowledge of peers to estimate the future information needs peers would require. The DDLS peer network comprising all peers is reorganized by using this estimation to improve ad-hoc resource sharing on the Web.

The remainder of the article is structured as follows. In Section 2, we present the enabling technologies for the DDLS. We describe in Section 3 the design issues of the DDLS. Reorganization that is intended to enhance resource discovery in the DDLS peer network is discussed in Section 4. Section 5 details simulations on semantic search and reorganization. Finally, we outline some conclusions and future work in Section 6.

2. BACKGROUND

This section first discusses resource sharing in hypermedia link services, or link services. This is followed by an introduction to P2P computing which has the potential to support resource sharing on the Web in an ad-hoc manner and a taxonomy of contemporary P2P systems with their strengths and weaknesses highlighted. Finally, we present the major forerunners of the DDLS, including the DLS from which the DDLS inherits its core concept and Microcosm TNG that demonstrates using a centralized P2P model in link services to facilitate resource sharing.

2.1 Resource Sharing in Link Services

The link service is a term first used by Pearl [1989]. Every OHS (e.g. Microcosm [Fountain et al. 1990], DHM [Grønbæk et al. 1993], Hyper-G [Andrews et al.

1995], Chimera [Anderson 1997] and HyperDisco [Wiil and Leggett 1997]) requires a link service of one form or another to manage links. This is because the open hypermedia model enables the maintenance and manipulation of links separately (in linkbases) from the documents they describe which fundamentally differs from the model of closed hypermedia systems, such as the Web.

Allowing links to be manipulated separately from documents makes it possible to describe resources and store the description with links referring to them in linkbases. Resource discovery is turned into a link discovery problem; link services address it by querying against linkbases, thus resource sharing becoming link sharing.

Typically, link services host a central service directory for link location. This makes link discovery simple but increases user dependence on a single link server and introduces single points of failure. Further, users may not be able to obtain updated links from others in a timely manner. We introduce in the following section P2P computing that is capable of removing these shortcomings and delivering mechanisms for supporting link services in a decentralized adhoc manner.

2.2 P2P Computing

P2P computing can be described as an overlay network in which a group of peers communicate using the same networking program. Each peer possesses equal capabilities taking actions independently of each other. By employing distributed resources, peers perform a common function in a decentralized manner. This paradigm, however, does not preclude centralization in some parts of the network. The P2P network aims to provide services even in face of unstable connectivity. There are three categories of the P2P network: centralized, unstructured, and structured.

2.2.1 *Centralized P2P.* A centralized P2P system, Napster for instance, has a central repository to maintain the information about resources in the network. Resource discovery in such systems involves a lookup in the central repository. A central peer or a group of dedicated peers coordinate updates to the information held in the repository. A centralized P2P system is susceptible to common denial-of-service attacks. It may also give rise to the network hot spot, which hinders the development of a scalable system using a centralized P2P model.

2.2.2 *Unstructured P2P.* Gnutella is possibly the most appealing and controversial unstructured P2P system as of today. As a P2P file-sharing application and protocol, it models the realistic world better than a structured P2P system (see Section 2.2.3) in that the placement of data objects is not subject to any knowledge of the network topology. Gnutella does not maintain an index of files available in the network. A search is performed by propagating queries between nodes and is terminated on the successful retrieval of documents or on achieving the termination conditions. An example condition is a TTL (Time-To-Live) tag specifying the number of hops a query can be relayed. Using flooding broadcast in Gnutella incurs an enormous number of messages and consumes excessive network resources.

Other examples of unstructured P2P include KaZaA, Routing Indices (RIs) [Crespo and Garcia-Molina 2002] and the hierarchy of resemblance (HR) search [Larsen and Bouvin 2004]. KaZaA uses specially designated supernodes with higher bandwidth connectivity to store information about the resources of peers and routes all queries to the supernodes, whereas RIs provide a distributed index scheme to facilitate document retrieval. Each peer has a local RI to maintain information on different topics of documents along each path to its neighbors which, in turn, collects information from the documents of its neighbors using the same mechanism. The HR search relied on a hierarchy data structure in which peers were ranked according to the previous search results. The well-known random walk technique was adopted to facilitate searching the hierarchy for distributed hypermedia structures.

2.2.3 *Structured P2P.* A typical structured P2P system, such as CAN [Ratnasamy et al. 2001], Chord [Stoica et al. 2001], Pastry [Rowstron and Druschel 2001] and Freenet-like GUIDs [Lukka and Fallenstein 2002], indexes the search space and uses DHTs (Distributed Hash Tables) to map keys onto nodes for efficient query routing. The adoption of DHTs assumes a highly structured system in which the P2P network topology is tightly controlled and the placement of data items is precisely determined. DHTs offer a very scalable solution to exact match queries as a lookup operation typically scales as $\Theta(\log N)$, given N nodes in the system. DHTs assume the uniqueness of keys and only return data items in a node whose identifier is exactly mapped from a specified key. A search for multiple data items by using DHTs involves searching for the items in a sequential manner and performing conjunctive or disjunctive operations on the result with the assistance of filter structures such as Bloom filters [Mullin 1990]. However, the relationships between items are thoroughly omitted during the search.

2.3 Major Forerunners of the DDLS

2.3.1 *DLS.* The first version of the DLS (Distributed Link Service) [Carr et al. 1995] was a Web-based hypermedia link service that satisfied a user's information needs by providing links that referred to the documents of interest. It had an associated main linkbase used by the link server that provided the server facilities. Additional linkbases were available that enabled the server to offer a range of different sets of links, known as contexts. The personalization feature of the DLS allowed each user to create their own linkbases. When the link server responded to a follow link request, it looked up all the records that shared the same source selection in the specified linkbases and decided whether the link might be followed in the current context, whereas in the Web environment, the same request may lead to the link being followed to only one destination rather than many.

To be precise, the DLS was more like a dynamic link service than a truly distributed link service as it maintained linkbases on the same server as that of the link resolver component. The proxy DLS [Carr et al. 1998] proposed a network model which introduced multiple link servers and facilitated the distribution of linkbases. However, neither the algorithm for query routing nor the details of how the proxy DLS dealt with a query that involved linkbases on multiple link servers were revealed. Further, De Roure et al. [2000] discussed query processing in a distributed context and demonstrated that directory services could be very useful within the link service infrastructure.

2.3.2 *Microcosm TNG.* Microcosm TNG (The Next Generation) [Goose 1997] was intended to facilitate distributed information sharing and organization. Like its predecessor Microcosm [Fountain et al. 1990], Microcosm TNG was designed to be P2P. Peers appeared in the form of user sessions in Microcosm TNG. A peer discovered services of interest by interrogating a message router which kept a record of registered service providers both inside and outside the domain. A domain typically involved one or more local/remote user sessions and a domain daemon process that provided a single point of contact within each domain and required each message router within the domain to register their network address. Microcosm TNG exposed a centralized P2P nature in that it maintained some form of central repository to publish linkbases and utilized a P2P communication model to share linkbases. However, Microcosm TNG did not address issues such as resource description and discovery arising in more dynamic environments.

3. THE DISTRIBUTED DYNAMIC LINK SERVICE

In this section, we describe the design issues of a prototype of the Distributed Dynamic Link Service (DDLS), including architecture, resource description, resource publishing, and resource discovery.

3.1 Design and Prototyping

3.1.1 *Architecture.* The DDLS is an extension of the DLS and is intended to support ad-hoc resource sharing among online users. Hence, we exclude a central service directory from its architecture. Figure 1 illustrates the DDLS architecture. Each client, or peer, has an associated link server. The link server is responsible for storing and manipulating the peer's personal linkbases (see Section 3.1.2), and more importantly, it handles link service requests from peers. The user interface captures a link service request, wraps it in an HTTP request, and forwards it to the peer's link server. The link server queries against peer's personal linkbases and also forwards queries involving public linkbases of others to related link servers. A peer locates the target link servers through the resource discovery mechanism presented in Section 3.1.4. Further, a follow link request will be forwarded by the link server to an HTTP proxy. The HTTP proxy then sends the request to an HTTP server to fetch the document of interest.

3.1.2 *Resource Description.* Linkbases, the repositories in which links are stored and manipulated, are the most essential resources in the DDLS. We address the issue of resource description at the linkbase level as a starting point. Because each linkbase maintains a list of links related to an abstract concept, we can characterize the linkbases of a peer based on their associated concepts in terms of a topic vector. Using the Resource Description Framework

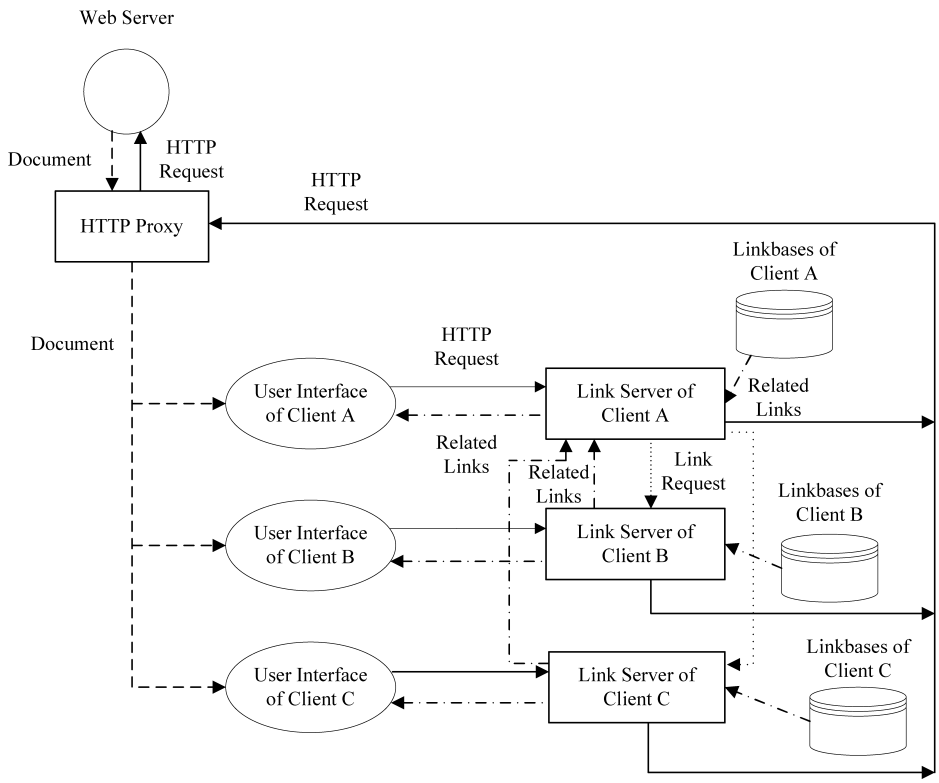

Fig. 1. The DDLS architecture.

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF xmlns:rdf=http://www.w3.org/1999/02/22-rdf-syntax-ns#
    xmlns:lb="http://www.ddls.com/rdf/linkbase-ns#">
    <rdf:Description about="http://www.ddls.com/linkbases/project/resource.xml">
        <rdf:type resource="http://www.ddls.com/rdf/linkbase-ns#Linkbase" />
        <lb:topic>design</lb:topic>
</rdf:Description>
</rdf:RDF>
```

Fig. 2. An example of using the RDF model to represent the DDLS linkbase.

(RDF) [Lassila and Swick 1999] to represent linkbases enables us to augment the linkbase description with related information, for example, location and type. Such information is encoded in sets of triples.

Figure 2 presents an example DDLS linkbase represented by the RDF model. It indicates that the resource being described is http://www.ddls.com/linkbases/project /resource.xml. The type of the resource is defined as another resource http://www.ddls.com/rdf/linkbase-ns#Linkbase. The primary content of this resource is design.

3.1.3 *Resource Publishing.* As a peer participates in the DDLS, it contacts a random set of peers already in the peer network, known as neighbors. The new participant informs each neighbor of its published topic information. It is

Table I. A Published Topic List in the Cache of Peer p_i

PeerID	α^{dist}	Topics
j	0	
k	2	D, E
m	5	G, H, I, J, K

assumed that the environment provides each peer with a capability to identify the semantic relationship between entities (see Section 3.1.4). All neighbors return related topics to the new peer. The latter subsequently takes advantage of this information to construct a published topic list (see Section 3.1.3) in cache.

The size of the cache is determined by a specified value or by a default of 128. In the list, each entry comprises three fields. The first field indicates all the neighbors that share α^{dist} (shown by the second field) related topics with the current peer. The contents of these topics are listed in the third field. Table I gives an example published topic list in the cache of peer p_i. It indicates that p_i shares no relevant topics with p_j. The neighbors that have semantically related topics with p_i are p_k and p_m, which have 2 and 5 relevant topics, respectively.

The published topic lists are mutable since the resources of peers may change over time. They are maintained in the cache of individual peers instead of in a central repository. This helps keep the freshness of information in the lists.

3.1.4 *Resource Discovery*. Resource discovery in the DDLS revolves around the location of desired linkbases. If the search mechanism relies on the standard keyword-based match, then only linkbases with matching syntax instead of matching concepts can be discovered. We present in this section our approach (which was partially reported in Zhou et al. [2003]).

3.2 Major Assumptions

The semantic search algorithm to be described is based on the following assumptions.

—*Capability to identify semantic relationships.* The environment provides each peer with a capability to identify the semantic relationship between topics representing the primary content of resources, such as being semantically related. The semantic similarity can be described in different ways and the existence of such a mechanism is assumed. It may exist in the form of a controlled vocabulary (e.g., ontologies) or may be based on inference logic, or otherwise.

—*Statically defined relationship among topics.* The search algorithm assumes that the semantic relationship between topics is statically defined and does not cater for environments in which the semantic relationship can be constantly redefined.

3.3 Query Mechanism: Topic Query and Associated Operations

The DDLS supports queries with conjunctive and disjunctive operations on query predicates, and in this work topics are used to define query predicates. A query expression is represented by conjunctive/disjunctive operations on

```
1:  <rdfq:rdfquery>
2:   <rdfq:From eachResource="http://www.ddls.com/
3:    linkbases/collaborative_environment_x/peer_linkbase">
4:    <rdfq:SELECT>
5:      <rdfq:Condition>
6:        <rdfq:and>
7:          <rdfsq:equal>
8:            <rdfq:Property name="lb:topic"/>
9:            <rdf:String>Topic A</rdf:String>
10:         </rdfsq:equal>
11:         <rdfsq:equal>
12:           <rdfq:Property name="lb:topic"/>
13:           <rdf:String>Topic B</rdf:String>
14:         </rdfsq:equal>
15:       </rdfq:and>
16:       <rdfq:or>
17:         <rdfsq:equal>
18:           <rdfq:Property name="lb:topic"/>
19:           <rdf:String>Topic C</rdf:String>
20:         </rdfsq:equal>
21:       </rdfq:or>
22:     </rdfq:Condition>
23:   </rdfq:SELECT>
24:  </rdfq:From>
25:</rdfq:rdfquery>
```

Fig. 3. The typical specification of DDLS topic queries.

groupings of topics. Figure 3 presents an example RDF topic query. The extensibility of RDF is utilized to define a tag <rdfsq: equal> that represents the state of being semantically related. Tags, such as <rdf:Integer> and <rdf:String>, can be employed to identify primitive data types. The query result needs to return all the linkbases semantically related to (*Topic A* and *Topic B*) or *Topic C*.

All topics in a conjunctive predicate need to be simultaneously satisfied by the description of linkbases, whereas those in a disjunctive predicate can be evaluated against the description of linkbases. The result of a disjunctive query in the DDLS is typically generated by merging the results of conjunctive subqueries. For instance, in Figure 3, the query is initially split into two subqueries, each of which contains a conjunct, wrapped in two separate messages. One subquery is constructed by statements from line 7–14 with italic typeface and the other from line 17–20. Both the query and its subqueries are assigned with Universally Unique Identifiers (UUIDs) [The Open Group 1997]. Suppose that the query identifier is 2fac1234-31f8-11b4-a222-002035b29092 which is inherited by both of its subqueries. The first subquery with subquery identifier 58f202ac-22cf-11d1-b12d-002035b29092 returns all the linkbases having related topics with both Topic A and Topic B. The second one with subquery identifier 5a389ad2-22dd-11d1-aa77-002035b29092 fetches all the linkbases possessing related topics with Topic C. Results of subqueries returned to the query originator will be merged and given the same query identifier 2fac1234-31f8-11b4-a222-002035b29092.

3.4 Distance-Based Semantic Search Algorithm

The DDLS semantic search algorithm is based on the concept of the distance vector, such as α^{dist}, as shown in Table I, that represents the proximity of resources of a pair of neighbors. Based on the proximity, a peer propagates queries to either some or all of its neighbors.

The details of the algorithm are described as follows. Any participating peer can initiate a semantic search query. The query is evaluated against the initiator's cached information to determine the distance between the query expression and the cached information about the neighbors. If the query evaluator finds a match, it routes the query to the associated peers. A match means there is an overlap between the query topics and the topics in an entry of the published topic list of the query evaluator. In case no match has been found, the query is propagated to all neighbors of the current query evaluator. The query is subsequently evaluated by each of its recipient peers. The number of hops for query propagation is limited by the lifetime of the query, expressed by a TTL tag as used in Gnutella. Query matches are directly routed back to the query initiator.

3.5 Discussions

We have presented the DDLS in support of ad-hoc resource sharing on the Web. The DDLS's capability of serving links on demand without the assistance of a central service directory relies on a distance-based semantic search mechanism. Unlike other related work on semantic search [Deerwester et al. 1990; Heflin and Hendler 2000; Li et al. 2002; Guha et al. 2003], the DDLS assumes the ability of identifying semantically-related terms that all related work possesses and focuses on an underlying mechanism that supports the semantic search in a P2P context that none of them is able to tackle.

In the DDLS semantic search algorithm, a query is propagated to all neighbors only if no match is found in the cache of the query evaluator. This should not be interpreted the same as in Gnutella where queries are always broadcast to all neighbors of the query recipient. On the one hand, query broadcasting is a compromise between reducing the consumption of network resources and extending the possibility of locating targets. On the other hand, as we will present in the following section, the DDLS peer network will perform reorganization whenever necessary, which enables peers to become neighbors of others sharing semantically-related resources. As a consequence, the time of query broadcasting can be reduced. This also explains why we consider reorganization as an integral part of the DDLS.

4. REORGANIZING THE DDLS PEER NETWORK

In this section, we define the concept of reorganization in the context of the DDLS peer network and introduce a data structure, query history, that plays an important role in facilitating reorganization. We then analyze resource supply and demand in the DDLS peer network and summarize the major criteria for reorganization resulting from this analysis. Finally, we propose two different techniques, the exponential decay function and the naive estimator[3], to assist peers in making decisions on choosing neighbors during reorganization.

4.1 An Introduction to Reorganization

In the context of the DDLS peer network, we define reorganization as an act of altering the virtual neighborhood of any peer in the network to optimize resource discovery. It can be triggered by the state change of any peer in the network including the (dis)appearance of a peer and resource/neighbor update to a peer. In this work, we are particularly interested in update-related reorganization.

The inherent organizational structures of P2P systems differ from one another. For instance, centralized P2P would not benefit from reorganization because resource discovery in such systems heavily relies on a central service directory and altering the network topology does not facilitate resource location. For structured P2P that adopts DHTs to index the search space for efficient lookup, reorganization bears no significance either. Unstructured P2P is perhaps the most promising system that would benefit from employing reorganization because it does not depend on a central directory and neither does it have a tight control over the network topology and the placement of resources. However, little work on reorganization in unstructured P2P has been reported.

Another reason for adopting reorganization in the DDLS is that the knowledge of resources owned by each peer is restricted to its neighbors and peers view a limited scope of the global information about resources in the system—reorganization can help broaden the peer's horizons.

The third reason is that, according to the DDLS semantic search algorithm, a query will be broadcast if no match can be found in the cache of the query evaluator. By making peers with semantically-related resources neighbors, the chance that a query can be satisfied by neighbors increases, thus reducing the time of expensive broadcast.

When reorganization occurs, involved peers can employ different reorganization techniques (see Section 4.4 and 4.5) to discover a new set of neighbors that are most qualified to help achieve the objective of reorganization and replace their current set of neighbors with the new one. These techniques rely on an important data structure, query history that maintains for peers the routing information of queries propagated from others. The next section introduces this data structure and related operations.

4.2 Query History

Query history is a collection of all queries a peer has encountered over a period of time. Peers may be able to predict the future information needs of others by analyzing query history which is subsequently accompanied by rational reorganization.

[3]We reported our preliminary study on naive estimator-supported reorganization in Zhou et al. [2004].

Query history is realized as a FIFO (First In First Out) queue. The oldest entry of the query history is discarded when the queue is full. Let T denote the set of all the topics. Each entry of query history includes three fields: the query identifer, query topics, and the arrival time of the query. The set of query identifiers is Q, the capacity of query history of peer p_i is h_i^{max}, and the set of arrival time of queries is A. Query history of p_i can be represented by $H_i = \{(q_i^m, h_i^m, a_i^m) | q_i^m \in Q, h_i^m \in T, h_i^m \times h_i^m \subseteq T, a_i^m \in A, 0 < m \leq h_i^{max}\}$.

4.3 Criteria and Metric

The knowledge of resource supply and demand in the DDLS peer network is essential for developing the reorganization techniques as reorganization aims at enhancing the performance of resource discovery. We present the two following criteria that result from an analysis of resource supply and demand in the DDLS and should be utilized to guide rational reorganization.

(1) Peers that share related topics should be incorporated into the same cluster[4].
(2) Peers should be situated in the vicinity of those that would accommodate potential information needs with high probability.

Criterion (1) indicates that peers with related topics should become neighbors since this allows a peer to identify others that accommodate similar information needs, or demand, conveyed by queries. Hence, the peer can forward subsequent related queries and avoid costly broadcast; key to the reduction of local broadcast is the local knowledge of the supply available at neighbors. Meanwhile, a peer can identify the potential demand of others from its query history. By taking advantage of this demand information, peers are able to choose their best neighbors during reorganization. This gives rise to criterion (2).

We employ the term *usefulness* to represent the relative extent to which a peer should become a neighbor of another peer during reorganization. Assume that candidate neighbor p_j (of p_i) publishes a topic list T_j. Let $\varepsilon_{i,j}$ be a metric that represents the information needs exhibited in the query history of p_i. Different techniques will be utilized to estimate the value of $\varepsilon_{i,j}$ in the following sections. Also, let $\eta_{i,j}$ denote the extent to which p_j would match the queries that p_i can satisfy:

$$\eta_{i,j} = \frac{|T_i \bigcap T_j|}{|T_i|}.$$

Let $\iota_{i,j}$ represent the usefulness of p_j with respect to p_i

$$\iota_{i,j} = \sqrt{(\kappa_1 \varepsilon_{i,j})^2 + (\kappa_2 \eta_{i,j})^2}, \tag{1}$$

[4]The term *cluster* is derived from the unsupervised clustering method that groups entities into clusters by the similarity of their features without any prior knowledge about the number of the clusters, which fits in with the DDLS clustering problem. However, because the resources of each peer are represented by a set of characteristic topics, a peer incorporated into more than one cluster would occur frequently.

where κ_1 and κ_2 are constant coefficients associated with query history and the cached topic information, respectively. The quantitative relationship of the significance between $\varepsilon_{i,j}$ and $\eta_{i,j}$ can be adjusted by assigning specific values to κ_1 and κ_2. Within the capacity of its cache, p_i keeps peers with the greatest value of ι as its new neighbors during reorganization and discards the rest.

In the following sections, we introduce the exponential decay function and the naive estimator that support reorganization to deliver an improved performance of resource discovery in the DDLS. The fundamental assumption behind both techniques is that the recent past will approximate the immediate future.

4.4 Exponential Decay Function-Based Usefulness Decision

A straightforward approach to distinguish the instances of query topics with various times of occurrence, is to allocate various weights to these instances. An exponential decay function $W(S(q))$ can satisfy the requirement. Let $W(S(q))$ be a weight function of $S(q)$ which is, in turn, a sequence function of an incoming query q. The following phenomena can be observed in query history to which an exponential decay function is applied.

- —More recent query topic instances are always awarded higher weights for their occurrences.
- —Two query topic instances that occurred more recently will have a bigger gap between their weight than another two (with the same distance in the sequence of query history entries) that occurred less recently.

Let h_i^{max} be the capacity of query history H_i of p_i. The exponential decay function takes the following form:

$$W(S(q)) = e^{-S(q)}.$$

For sequence function $S(q)$, the following equations hold.

$$S\left(q_i^1\right) = 1, S\left(q_i^2\right) = 2, \ldots, S\left(q_i^{h_i^{max}}\right) = h_i^{max},$$

where $q_i^1, q_i^2, \ldots, q_i^{h_i^{max}}$ are a sequence of incoming queries ordered by the arrival time with q_i^1 being the most recent incoming query.

Suppose p_i needs to decide how useful neighbor p_j is. For a time interval I, the metric $\varepsilon_{i,j}$ that represents the information needs exhibited in the query history of p_j during the interval, takes into account all query instances in the query history of p_i whose topics are semantically subsumed by p_j's topics. Therefore,

$$\varepsilon_{i,j} = \sum W(m) = \sum e^{-m},$$

where $0 < m \leq h_i^{max}$ and $a_i^1 - I \leq a_i^m \leq a_i^1$.

Again, let $\eta_{i,j}$ be the extent to which that p_j would match the queries that p_i can satisfy and $\iota_{i,j}$ represent the usefulness of p_j with regard to p_i.

$$\iota_{i,j} = \sqrt{\left(\kappa_1 \sum e^{-m}\right)^2 + (\kappa_2 \eta_{i,j})^2}. \tag{2}$$

4.5 Naive Estimator-Based Usefulness Decision

The foundation of the naive estimator [Rosenblatt 1956] is that, for any given h and n independent observations $X_1, X_2, \ldots, X_n$ from the random variable X, the probability $P(x - h < X < x + h)$ can be approximated by the proportion of the samples falling in the interval $(x - h, x + h)$. Thus the naive estimator $\hat{f}_h(x)$ for the estimation of density value $f(x)$ at point x is defined as

$$\hat{f}_h(x) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{h}w\left(\frac{x - X_i}{h}\right),$$

with h being a small number and the weight function defined as

$$w(x)\begin{cases}\frac{1}{2} & \text{if } |x| < 1\\ 0 & \text{otherwise.}\end{cases}$$

The naive estimator is a nonparametric approach [Silverman 1986] in that less rigid assumptions, for example, the density function underlying the data, are made about the distribution of the observations. It is the observed data that is crucial in determining the estimate of $f(\bullet)$. Inspired by this feature, we realized that in the DDLS the distribution of query topics can be estimated based on the overall view of peers, while the probability of query topics at a single peer can be approximated by the peer's local view.

The purpose of introducing query history is to perform an informal investigation into the properties of queries, and guide reorganization of the peer network using the properties of queries revealed. In the DDLS, the probability of query topics is an important property that needs to be explored for predicting the future information needs. The naive estimator can take advantage of the query history maintained by individual peers to approximate the future information needs they would require.

Suppose the probability of topics in the query history of peer p_i can be depicted by function $f_{i,h}(t)$ of a discrete random variable t, where t denotes the least index of the same set of related topics in history entries. For example, if T_a and T_b denote the topics of the 1st and 5th entries in query history and both share the same set of related topics, then the observations of t for both entries will be 1 instead of 5. Using the naive estimator, the estimate of probability of topics at t is

$$\hat{f}_{i,h}(t) = \frac{1}{n}\sum_{k=1}^{n}\frac{1}{h}w\left(\frac{t - T_k}{h}\right),$$

with $h = 0.5$. It should be stressed that $\hat{f}_{i,h}(t)$ takes into account all entries in the query history of p_i no matter when they arrived. This is contrary to what occurs in exponential decay function-based usefulness decision.

Assume that neighbor p_j of p_i publishes its topic list T_j. Let $\varepsilon_{i,j}$ be the estimate of the probability of topics in T_j in future queries encountered by p_i.

$$\varepsilon_{i,j} = \sum \hat{f}_{i,h}(t).$$

$\varepsilon_{i,j}$ considers the estimate at all t where the topics of history entries are semantically subsumed by topics in T_j. Let $\iota_{i,j}$ represent the usefulness of p_j with regard to p_i and $\eta_{i,j}$ be the extent to which that p_j would match the queries that p_i can satisfy.

$$\iota_{i,j} = \sqrt{\left(\kappa_1 \sum \hat{f}_{i,h}(t)\right)^2 + (\kappa_2 \eta_{i,j})^2}. \tag{3}$$

5. SIMULATION

Due to the large scale of the proposed DDLS peer network and the complexity of its resource discovery problem, in this section we demonstrate, through a series of simulations the performance of the semantic search algorithm and the performance improvements brought in by employing reorganization.

5.1 Topic Distribution

The DDLS search mechanism locates semantically-related resources. Hence, the distribution of individual topics that represent the primary content of resources is not of interest. Instead, topics are grouped by semantics and the distribution of such topic groups are what should be utilized to study the DDLS search. Hereafter, topic popularity (or probability), which will be defined later, refers to the popularity (or probability) of topic groups each of which has distinct semantics unless indicated otherwise.

We first investigate the Zipf's distribution of topics. Zipf's law [1949] is named after the Harvard linguistic professor George Kingsley Zipf (1902–1950). It states that the frequency of occurrence of some event (P), as a function of the rank (i) that is determined by the frequency of occurrence, is a power-law function $p_i \propto \frac{1}{i^\alpha}$ with α close to unity. Zipf's distribution has been demonstrated to characterize the use of words in a natural language, for instance, English.

The term *topic popularity* represents how popular a topic/topics is/are in terms of the number of peers holding it/them. Let t_i be the topic popularity of the i-*th* topic in a Zipf's distribution.

$$t_i \propto \frac{1}{i^\alpha},$$

where $\alpha = 1$.

To compare against Zipf's distribution, the uniform distribution of topics was chosen in which each topic is shared by the same number of peers. The term *topic probability* denotes the percentage of peers that possess a topic/topics compared to all peers in the system. Let t_i be the topic probability of any topic in a uniform distribution,

$$t_i = C,$$

where C is a constant.

5.2 Metrics and Issues

It can be complicated to evaluate algorithm performance against a number of metrics, for instance, search speed and accuracy, the number of messages sent, system load, and resource consumption. In the context of the DDLS, performance issues are primarily measured by the following metrics.

—Hops: delay in finding all answers as measured in the number of hops.
—Recall: the fraction of matches that can be found.
—Broadcast rate[5]: the time of broadcast carried out by all peers to propagate queries over a period of time.

Elements that may have an effect on hops, recall, and broadcast rate will be investigated. We conjectured that exploring answers to the following questions would be helpful in understanding the search algorithm and hence devised the simulations in Section 5.3 and Section 5.4.

(1) What is the basic behavior of the semantic search in terms of the aforementioned metrics?
(2) What is the relationship between the amount of information a peer should cache about its neighbors and the search performance?
(3) Does the topic distribution have an impact on the search performance?
(4) Should the peer network remain unchanged or be reorganized to improve the search performance? If so, what techniques should be utilized to guide the reorganization process?

5.3 Simulation on Semantic Search

5.3.1 *Semantic Search.* The simulations described in this section[6] reveal the behavior and performance of resource discovery in the DDLS that employs the distance-based semantic search algorithm. Semantic search was carried out in peer networks in which topics follow a Zipf's distribution or a uniform distribution. We executed 20 runs for each simulation and averaged the data collected.

Experimental settings (first). We define the cache rate as the percentage of peers whose topic information is in cache compared to all peers in the system. This first experiment was intended to explore the relationship between the cache rate and hops.

We performed the experiment involving 100 peers[7] in a controlled environment in which the distribution of topics was kept constant throughout the experiment and the number of peers in the system was restricted. Each peer was associated with a certain cache rate and can randomly choose a list of topics from a global list of 100 entities, ensuring the topics of all peers followed a specified distribution. The cache rate was varied from 1%, 2%, to 90%. The fifteenth most popular topic, shared by 5 out of 100 peers in the system, was chosen as the query topic for the experiment with Zipf's distribution. A topic with the topic

[5]This metric can be used to estimate the consumption of network resources during resource discovery, for instance, the number of potential messages generated in resolving a query.
[6]Due to space limit, we present only experiments on single topic search and refer the readers to Zhou [2004] for more details on multiple topic search that exhibits qualitatively similar behaviors.
[7]At the current stage, we are interested in exploring the basic behavior and performance of the DDLS and will investigate in future work its scalability compared to other systems.

Table II. Cache Rate, Hops and Recall Resulting from Searching for a Topic Shared by 5 Out of 100 Peers (Zipf's Distribution)

Cache Rate	Hops	Recall
1%	0.01	1%
2%	7.36	80%
3%	5.85	100%
4%	5.09	100%
5%	4.41	100%
10%	2.83	100%
20%	2.05	99%
30%	2.00	98%
40%	2.01	99%
50%	1.98	100%
60%	1.92	100%
70%	1.84	100%
80%	1.63	100%
90%	1.40	100%

Table III. Cache Rate, Hops and Recall Resulting from Searching for a Topic Shared by 5 Out of 100 Peers (Uniform Distribution)

Cache Rate	Hops	Recall
1%	2.47	9%
2%	8.86	80%
3%	6.14	100%
4%	4.91	100%
5%	4.13	100%
10%	3.09	100%
20%	2.06	100%
30%	1.98	98%
40%	2.00	99%
50%	1.96	100%
60%	1.98	100%
70%	1.87	100%
80%	1.66	100%
90%	1.46	100%

probability of 5%, that is, shared by 5 out of 100 peers, was randomly chosen from the global list to formulate a query for the experiment with the uniform distribution.

Discussion. The results in Table II and Table III show that, regardless of the distribution the topic in a query is associated with, the number of hops is inversely proportional to the cache rate. It is observed that except the cases in which the cache rate equals 1% or 2%, that is, each peer only caches the topic information from one or two of its neighbors, the resource discovery mechanism can lead to a satisfactory recall (at least 98%) within the experimental settings. The cache rate at 1% and greater only guarantees that each peer is aware of at

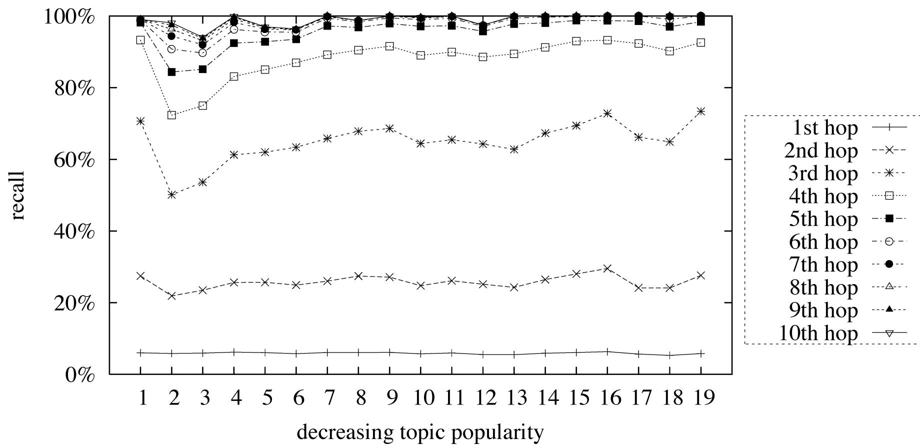

Fig. 4. Recall at successive hops (Zipf's distribution).

least another peer (neighbor), whereas it is not assured that each peer is known by at least another peer. Therefore, the recall cannot always reach 100%.

Experimental setting (second). The second experiment aimed to explore the properties of the semantic search algorithm of the DDLS in which topics follow Zipf's distribution.

The experiment was carried out over the first nineteen most popular topics. In a peer network consisting of 100 peers, Zipf's distribution of 100 topics yields 19 bands, each of which is occupied by topics that are shared by the same number of peers. The cache rate was kept at 5% throughout the experiment, and all the other experimental settings were retained as in the first experiment. To ensure each peer has at least one neighbor, the cache rate should be 1% or greater. However, it is shown that a very low cache rate can result in an unacceptable recall, for instance, 1% in Zipf's distribution and 9% in uniform distributions, which does not demonstrate the typical behavior and performance of the semantic search but represents the extreme case. Hence, this experiment and others presented later use 5% as the cache rate because such a relatively low cache rate is more realistic for a peer network that allows for a wide range of the number of peers.

Discussion. In search of topics with different popularities in Zipf's distribution, the DDLS delivers the recall at successive hops as plotted in Figure 4. It is shown that the second most popular topic shared by 50% of all peers is accompanied by the minimum recall at almost every hop. The finding indicates that a search for that topic will lead to more hops to achieve a certain level of recall compared to a search for any other topic. Table IV further demonstrates this with the most hops produced by a search for the second-most popular topic. Thereafter, the hops are in proportion to topic popularity, with the least-popular topic resulting in the least hops. This phenomenon can be explained as follows. The number of hops is subject to the probability of the query topic. On the one hand, the discovery of a certain number of instantiations of a

Table IV. Hops Resulting from Semantic Search for Topics that Follow Zipf's Distribution

popularity	1	2	3	4	5	6	7
hops	5.47	8.10	6.33	5.84	5.20	5.01	5.00
recall	99%	99%	94%	100%	97%	96%	100%
popularity	8	9	10	11	12	13	14
hops	4.74	4.71	4.85	4.75	4.60	4.70	4.44
recall	99%	100%	99%	100%	98%	100%	100%
popularity	15	16	17	18	19		
hops	4.22	4.01	3.96	3.77	3.06		
recall	100%	100%	100%	100%	100%		

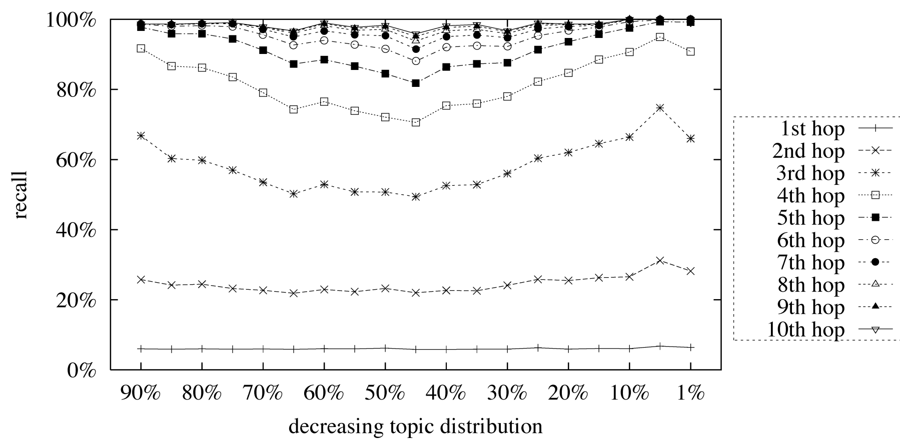

Fig. 5. Recall at successive hops (uniform distributions).

topic with low probability results in more hops. On the other hand, according to the search algorithm (see Section 3.1.4), low probability of the query topic yields low probability of overlap between peers, thus triggering a higher broadcast rate which implies less hops to locate the same number of instantiations. Let x be the probability of a query topic and $f(x)$ be the number of hops: $f(x) = d(x) * b(x)$. $d(x)$ and $b(x)$ are associated with the two aspects as previously analyzed with $d(x)$ as a decreasing function and $b(x)$ as an increasing function of x over [0, 1]. Because $f(x)$ is continuous over [0, 1] and $f(0) = f(1) = 0$, there must be at least a single point x at which $f(x)$ has its maximum.

Experimental settings (third). The third experiment was set up to investigate the properties of the semantic search algorithm of the DDLS in which topics follow uniform distributions.

We carried out the experiment with the topic probability ranging from 1%, 5%, 10% to 90%. Again, the cache rate was kept at 5% throughout the experiment, and all the other experimental settings were kept as in the first experiment.

Discussion. Figure 5 shows the recall at successive hops in search of topics with different probabilities in uniform distributions. Within the first 10 hops, a search for topics with a probability of 45% yields the minimum recall at almost

Table V. Hops Resulting from Semantic Search for Topics that Follow Uniform Distributions

probability	90%	85%	80%	75%	70%	65%	60%	55%	50%	45%
hops	5.51	6.04	6.04	6.32	6.81	7.59	7.66	8.28	7.72	7.83
recall	99%	99%	99%	99%	98%	97%	99%	98%	99%	97%
probability	40%	35%	30%	25%	20%	15%	10%	5%	1%	
hops	6.98	6.60	6.16	5.79	5.44	5.00	4.88	4.03	3.08	
recall	99%	99%	97%	99%	99%	99%	100%	100%	100%	

every hop. This phenomenon is analogous to the one in Figure 4 that the search for the second-most popular topic (shared by 50 out of 100 peers) results in the minimum recall at practically every hop. Table V reveals that the topic probability of 55% is related to the most hops. The number of hops increases before the topic probability reaches 55% and decreases thereafter. Contrary to the experiment with topics from Zipf's distribution (see the second experiment) in which 50% is the turning point for all observations, this experiment shows that the turning point exists in the range of [45%, 55%]. We speculated that both results should be consistent with each other, and the inconsistency in the results is due to the limited experimental conditions.

5.3.2 *Summary.* The experiments presented in this section investigated the DDLS resource discovery for topics following Zipf's distribution and uniform distributions, the potential distributions if topics are sorted by popularity. The main metrics in performance evaluation include hops, recall, and broadcast rate which have been employed to describe the following principal findings.

(1) When the cache rate is greater than 2%, the resource discovery mechanism can lead to a satisfactory recall (at least 98%).
(2) The number of hops is inversely proportional to the cache rate (except the extreme cases such as 1% in the simulation).
(3) In search of a topic with a probability less than 50%, the number of hops is inversely proportional to the topic probability/popularity, whereas in search of a topic with a probability more than 50%, the number of hops is proportional to the topic probability/popularity.

Further, we observed that the broadcast rate is inversely proportional to topic probability/popularity.[8] This is because a less popular query topic indicates a lower probability of overlap between peers, thus triggering a higher broadcast rate to locate all the instantiations of the topic.

The simulation result has given an answer to the first three questions raised in Section 5.2, that is, the anticipated behavior of the semantic search, the impact of the cache rate, and resource (or topic) distribution on the search performance. The last question that relates to reorganization of the peer network is left to be explored in the following section.

[8]The experimental data is omitted and we refer interested readers to Zhou [2004].

5.4 Simulation on Reorganization

Simulation on reorganization supported by the exponential decay function and the naive estimator, respectively, is described in this section. Both reorganization techniques are subsequently discussed and compared.

5.4.1 *Exponential Decay Function Supported Reorganization (EDFSR).* This section presents the simulation that investigates the behavior of reorganization supported by the exponential decay function. Particularly, the simulation explores the relationship between query history and the overlap information in usefulness decision and examines their individual significance for reorganization.

Experimental settings. The settings remain the same as in the second experiment of the semantic search. Moreover, a set of query topics that followed Zipf's distribution[9] was constructed and each query (482 instantiations in total derived from the global list of 100 entities) chose a single topic from this set. The capacity of query history was h. The experimental procedure is described as follows.

(1) Over a time interval I, q ($q \geq h$) queries are issued. A snapshot of query history of each peer and the topology of the peer network is maintained.
(2) Another q queries are issued over I in the peer network with the same topology as that maintained in step (1). The snapshot of the queries and the query initiators is kept.
(3) Based upon the query history and the topology maintained in step (1), the peer network is reorganized since a certain percentage, known as the updating rate *u.r.*, of all peers launch an update to their resources (which also results in an update to the topic information about the resources). For the sake of simplicity, these peers do not practically update topics in simulation but only choose their new neighbors in terms of usefulness ($\kappa_1 = 0$ and $\kappa_2 = 1$). The query initiators kept in step (2) issue the same queries as in step (2) over I.
(4) Repeat step (3) with pairs of values for κ_1 and κ_2: (0.01, 1), (0.1, 1), (1, 1), (10, 1), (100, 1) and (1, 0), respectively.

Utilizing the query history generated in step (1), step (2), (3) and (4) were set up to examine the hop reduction in various environments, including that without reorganization (step (2)) and those with reorganization (step (3) and (4)).

Discussion. Figure 6 shows the hop reduction that can be achieved by using different values for the exponent in the exponential decay function. The greater the absolute value of the exponent is, the steeper the slope of the curve that represents the function would be. This feature of the exponential decay function indicates that, among others used in the experiment, $f(m) = e^{-2m}$ would result in the greatest difference between the weights allocated to a pair of query history entries. The simulation result reveals that $f(m) = e^{-\frac{m}{500}}$ yields

[9]Studies show the presence of Zipf's law in Gnutella and Web queries [Breslau et al. 1999; Sripanidkulchai 2001].

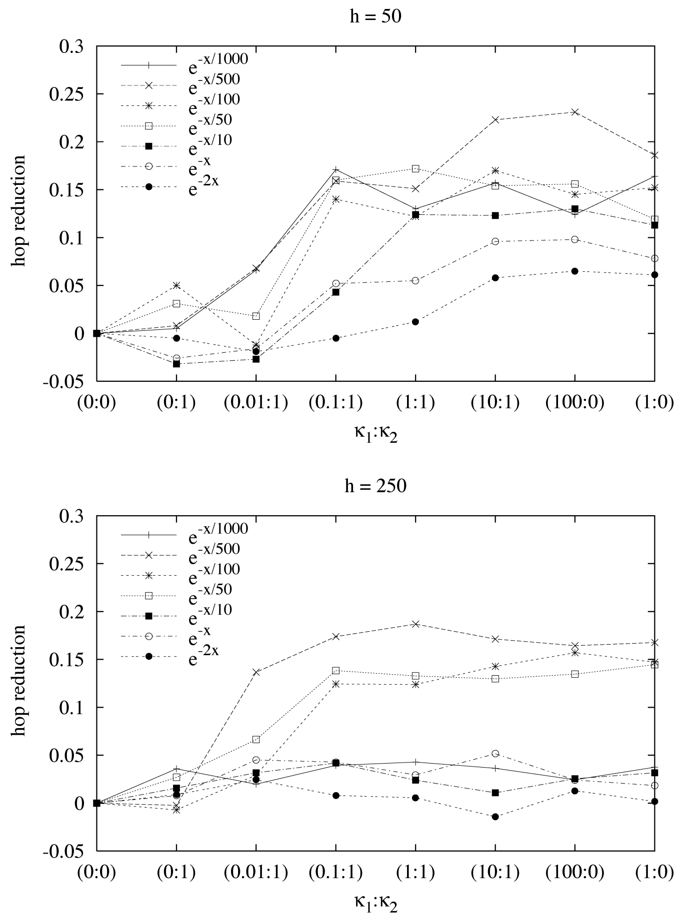

Fig. 6. Hop reduction in EDFSR that uses query history of different capacities (50 and 250).

the greatest hop reduction in most cases. However, the curve associated with $f(m) = e^{-\frac{m}{1000}}$ in the figure indicates that merely increasing the exponent does not necessarily lead to a greater reduction in hops. This is also supported by the result of the simulation conducted with the capacity of the query history equal to 250—conducting the simulation with a different history capacity is done to explore the frequency with which queries should be captured to facilitate reorganization.[10] Figure 6 demonstrates that, with an increased capacity (250) of the query history, the greatest hop reduction is accomplished by EDFSR associated with $f(m) = e^{-\frac{m}{500}}$ in most cases.

Figure 7 captures the impact that the updating rate has on hops. It can be seen that the more peers that carry out an update, the more hops are needed to discover all targets. We plot the hop reduction resulting from EDFSR that uses different updating rates in Figure 8. Only a relatively low updating rate,

[10]Figures 7 through 11 have a companion one that presents the result of simulations using h = 250, see Zhou [2004].

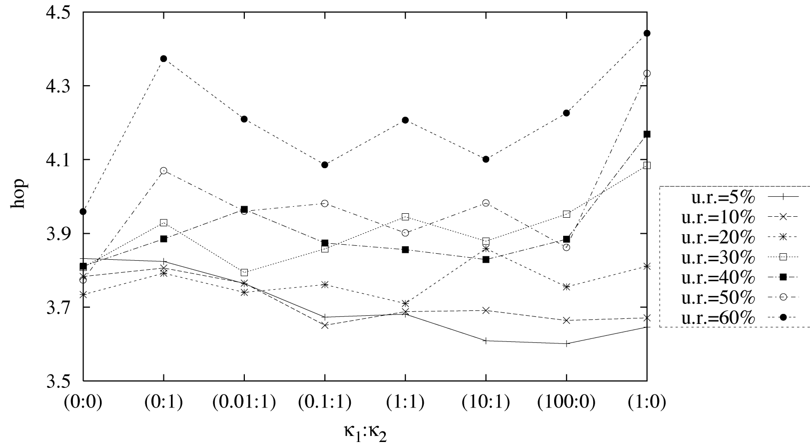

Fig. 7. Hops in EDFSR, $f(m) = e^{-\frac{m}{500}}$, $h = 50$.

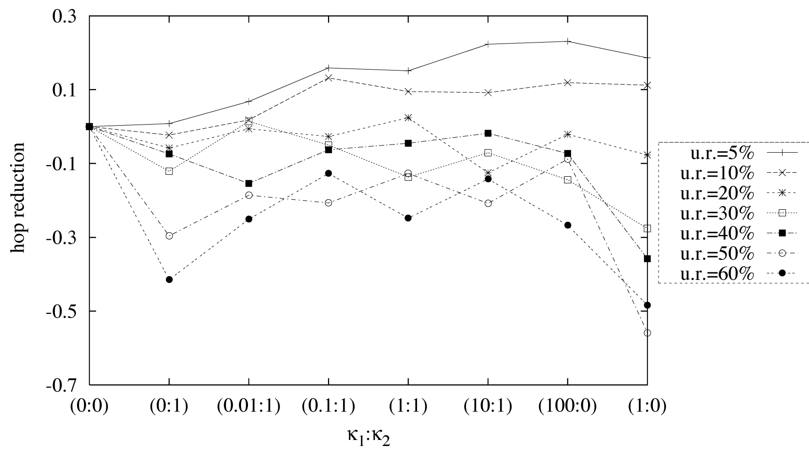

Fig. 8. Hop reduction in EDFSR, $f(m) = e^{-\frac{m}{500}}$, $h = 50$.

5% and 10% for example, gives rise to a reduction in hops. When the updating rate exceeds 20%, EDFSR does not necessarily lead to any hop reduction.

The recall and the broadcast rate delivered by adopting EDFSR can be seen to in Figure 9 and Figure 10. The curve associated with the updating rate of 60% is under all the rest in both subfigures, which indicates that one peer network with a higher updating rate (such as 60%) incurs a lower recall as well as a lower broadcast rate than another with a lower updating rate (such as 5%, 10%, and 30%). Moreover, an obvious phenomenon in the figure is that the recall is primarily proportional to the broadcast rate. In combination with Figure 7, one would discover that the relationship among the recall (r), hops (h), the broadcast rate (b), and the updating rate (u) across different combinations of κ_1 and κ_2, can be simply depicted by

$$r * u = C_1, \; b * u = C_2, \; h/u = C_3, \tag{4}$$

where C_1, C_2 and C_3 are constants related to ($\kappa_1 : \kappa_2$).

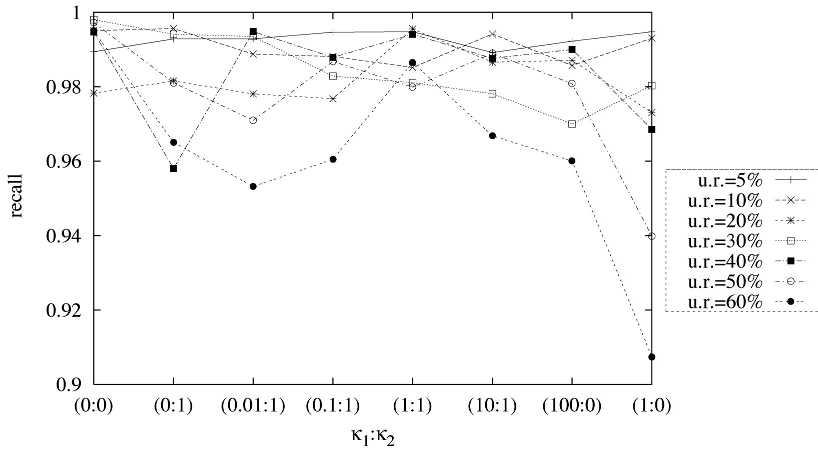

Fig. 9. Recall in EDFSR, $f(m) = e^{-\frac{m}{500}}$, $h = 50$.

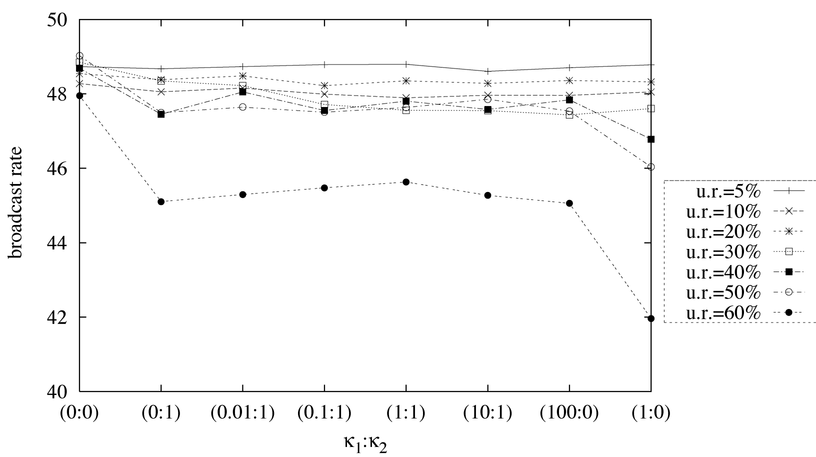

Fig. 10. Broadcast rate in EDFSR, $f(m) = e^{-\frac{m}{500}}$, $h = 50$.

5.4.2 *Naive Estimator Supported Reorganization (NESR).* The simulation in this section was carried out to explore the properties of reorganization supported by the naive estimator. Also, it helps understand the quantitative relationship between query history and the overlap information in usefulness decision of re-organization.

Experimental settings. The experimental settings remain the same as in Section 5.4.1 except that usefulness decision is enabled by the naive estimator.

Discussion. It is observed in Figure 11 that the impact from query history is predominant in hop reduction when the updating rate is relatively low, for instance, 5% and 10% in the experiment. As the updating rate increases (see the curves associated with the updating rate equal to 40%), the overlap information becomes more influential on hop reduction than query history.

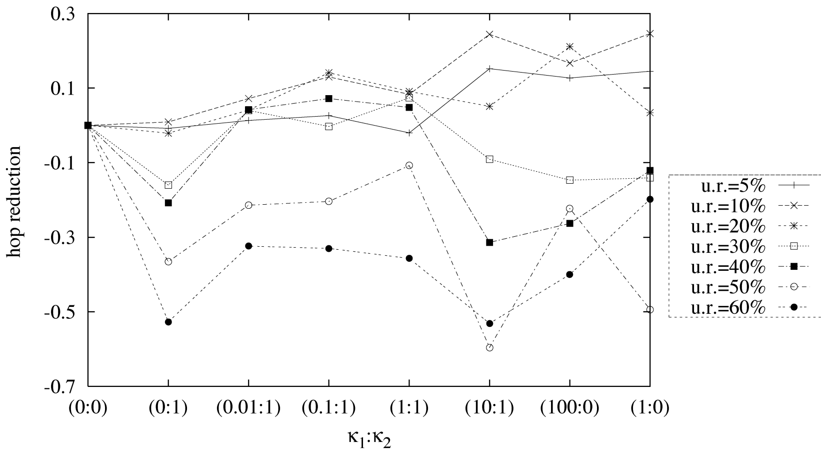

Fig. 11. Hop reduction in NESR, $h = 50$.

This experiment also reveals that, compared to EDFSR, NESR is applicable to a more dynamic peer network, that is, a peer network with a higher updating rate in terms of hop reduction. Recall that, as shown in Figure 8, a peer network with an updating rate greater than 20% results in no hop reduction when using EDFSR. Figure 11 demonstrates that NESR can boost the threshold up to 40%.

We observed similar patterns in NESR (figures are omitted) as those illustrated by Figures 7 through 10. In the DDLS peer network using NESR, the more peers that conduct an update, the more hops are needed in search of all targets. A higher recall is always accompanied by a higher broadcast rate which indicates that the ratio of the recall to the broadcast rate is nearly constant. As in EDFSR, Equation (4) holds across different combinations of κ_1 and κ_2 in NESR.

5.4.3 *Comparison Between EDFSR and NESR.* EDFSR and NESR share the same assumption that the recent past will approximate the immediate future. They both rely on the observation of queries in the past to estimate future information needs. If comparing each figure of simulation on EDFSR with its counterpart of simulation on NESR, one would discover qualitatively similar patterns from both. We found that for both techniques the greater the updating rate, the more the performance of resource discovery (except the broadcast rate) deteriorates. This situation can be partly ameliorated by reducing the capacity of the query history which translates to decreasing the time interval during which queries are captured. Hence, the updating rate becomes lower.

Meanwhile, EDFSR and NESR greatly differ from each other in terms of foundations, operational means, and application domains (see Table VI).

5.4.4 *Summary.* We have evaluated and confirmed the efficiency of proposed reorganization techniques (EDFSR and NESR) through a series of simulations in which recall, hops, and broadcast rate were utilized as metrics.

Table VI. Comparison Between EDFSR and NESR

	EDFSR	NESR
foundations	recency and frequency	a variant of frequency
operational means	Queries with the same frequency may be allocated different significance.	The equal frequency of a pair of queries in query history dictates their equal significance.
applicability	updating rate up to 20%	updating rate up to 40%

The main findings for both techniques are similar in pattern and are summarized as follows.

(1) The more updating peers, the more hops are needed in search of all targets.
(2) The impact from query history is predominant in hop reduction when the updating rate is relatively low (not more than 20% in EDFSR and 40% in NESR). Otherwise, the overlap information becomes more influential than the query history, and, in this case, reorganization may not necessarily lead to a better performance (except a reduced broadcast rate).
(3) One peer network with a higher updating rate (such as 60%) incurs a lower recall, a lower broadcast rate, and more hops than another with a lower updating rate (such as 5%, 10%, and 30%).

We demonstrated that the exponential decay function compared to the naive estimator is applicable to a relatively less dynamic peer network (see finding (2)). However, it should be pointed out that the simulations adopted Zipf's distribution for query topics, a pattern widely observed in large scale distributed systems such as the Web and Gnutella. This is a generalization of query distribution over the course of days or even months and does not capture any time-related feature of queries such as during which period a certain query was the most popular one. This explains why Zipf's distribution of query topics favors NESR. The real potential of EDFSR will only be fully exploited when work is conducted to examine the typical pattern of query topics in hypermedia link services in terms of a combination of both recency and frequency. Moreover, the same finding shows that, due to the unpredictable dynamics of the network at the construction stage, reorganization techniques should not be applied. Hence, it is reasonable for peers to randomly choose neighbors when the peer network is initially established.

We evaluated the performance of resource discovery by using the metrics defined in Section 5.2, respectively, while a comprehensive evaluation calls for a simulation result measured by some utility function that consists of a combination of these metrics. We left this to be accomplished in response to different available computational conditions and requirements for reorganization.

6. CONCLUSIONS AND FUTURE WORK

We began this work with a vision of implementing a hypermedia link service that is able to facilitate ad-hoc resource sharing between a community of Web users. This has been accomplished by using the open hypermedia paradigm to maintain and present resources, RDF to encode information about the resources, the clustering technique to group resources and form the information space, a semantic search mechanism to discover resources, and reorganization techniques to enhance the performance of resource discovery.

The DDLS approach is not limited to application to hypermedia link services. Other distributed systems that require support to operate in an ad-hoc environment may benefit significantly from growing into the problem space addressed by this work. For instance, locating other peers in a collaborative P2P system or any specific resource on the Grid [Foster et al. 2001] may convert the single attribute-based DDLS semantic search to multiple attribute-based search. This could involve associating tunable weight with each attribute according to its importance.

The design of the DDLS was carried out with an assumption that peers are provided with the capability of identifying semantic relationships between concepts. Essentially, an ontology is one of the mechanisms that enable this.[11] In future work, we will either utilize, merge, or extend existing ontologies and establish an approach to use ontologies in a more rigorous and explicit manner. Also, there is no previous research on the pattern of resource and query distribution in hypermedia link services. Without that knowledge, we can only speculate on the potential distribution based on observations from other large-scale distributed systems. This issue is also under investigation.

Further, we envisioned a natural extension of this work to address the scalability issue in triple stores built specifically for the Semantic Web [Berners-Lee et al. 2001]. This is because there is a demanding need for scalable techniques to remove performance bottlenecks in existing centralized triple stores of large scale, and we have long believed that introducing semantics awareness into resource discovery in unstructured P2P network, as we did in this work, could enable better performance to be delivered than that obtained by using canonical techniques such as flooding-based Gnutella and random walk. Moreover, since both the Semantic Web and the open hypermedia allow resources to be described by attached meaning (in terms of metadata), it is feasible that we apply the DDLS approach to engineer large-scale triple stores. At the time of writing, we have developed a very close variant of the DDLS semantic search algorithm and demonstrated its superiority over others in enhancing the scalability of triple stores. This complements our scalability study of the DDLS in this work. We will present the details in a forthcoming article.

REFERENCES

ANDERSON, K. M. 1997. Integrating open hypermedia systems with the World Wide Web. In *Proceedings of the ACM Conference on Hypertext*. ACM Press, 157–166.

ANDREWS, K., KAPPE, F., AND MAURER, H. 1995. Serving information to the Web with Hyper-G. In *Proceedings of the 3rd International World Wide Web Conference*. Elsevier North-Holland, Inc., Darmstadt, Germany.

[11]In addition to RDF and RDF Schema [Brickley and Guha 2004], other technologies such as OWL (Web Ontology Language) [Smith et al. 2004], a vocabulary extension of RDF and RDF Schema, and OWL-S (Ontology Web Language for Services) [Martin et al. 2004] which is built upon OWL, can be utilized instead.

BERNERS-LEE, T. 1996. The World Wide Web—Past, present and future. *J. Digital inform. 1*, 1.

BERNERS-LEE, T., HENDLER, J., AND LASSILA, O. 2001. The Semantic Web. *Scientific American 284*, 5, 34–43.

BRESLAU, L., CAO, P., FAN, L., PHILLIPS, G., AND SHENKER, S. 1999. Web caching and Zipf-like distributions: Evidence and implications. In *Proceedings of IEEE INFOCOM'99*. IEEE Computer Society, 126–134.

BRICKLEY, D. AND GUHA, R. 2004. RDF vocabulary description language 1.0: RDF schema. W3c recommendation, World Wide Web Consortium.

CARR, L., DE ROURE, D., HALL, W., AND HILL, G. 1998. Implementing an open link service for the World Wide Web. *WWW J. 1*, 2, 61–71.

CARR, L. A., DE ROURE, D. C., HALL, W., AND HILL, G. J. 1995. The distributed link service: A tool for publishers, authors and readers. *WWW J. 1*, 1, 647–656.

CLARK, D. 2001. Face-to-face with peer-to-peer networking. *Comput. J. 34*, 1, 18–21.

CRESPO, A. AND GARCIA-MOLINA, H. 2002. Routing indices for peer-to-peer systems. In *Proceedings of the 22nd International Conference on Distributed Computing Systems*. IEEE Computer Society, 23–34.

DE ROURE, D. C., WALKER, N. G., AND CARR, L. A. 2000. Investigating link service infrastructures. In *Proceedings of the 11th ACM Conference on Hypertext and Hypermedia*. ACM Press, 67–76.

DEERWESTER, S., DUMAIS, S. T., FURNAS, G. W., LANDAUER, T. K., AND HARSHMAN, R. 1990. Indexing by latent semantic analysis. *J. Amer. Soc. Inform. Sci. 41*, 6, 391–407.

FOSTER, I., KESSELMAN, C., AND TUECKE, S. 2001. The anatomy of the grid: Enabling scalable virtual organization. *Int. J. High Perform. Comput. Appl. 15*, 3, 200–222.

FOUNTAIN, A. M., HALL, W., HEATH, I., AND DAVIS, H. C. 1990. MICROCOSM: An open model for hypermedia with dynamic linking. In *Proceedings of the European Conference on Hypertext (ECHT'90)*, A. Rizk, N. Streitz, and J. Andre, Eds. Cambridge University Press, 298–311.

GOOSE, S. 1997. A framework for distributed open hypermedia. Ph.D. thesis, University of Southampton.

GRØNBÆK, K., HEM, J. A., MADSEN, O. L., AND SLOTH, L. 1993. Designing Dexter-based cooperative hypermedia systems. In *Proceedings of the 5th ACM Conference on Hypertext*. ACM Press, 25–38.

GUHA, R., MCCOOL, R., AND MILLER, E. 2003. Semantic search. In *Proceedings of the 12th International Conference on the World Wide Web*. ACM Press, 700–709.

HEFLIN, J. AND HENDLER, J. 2000. Searching the Web with SHOE. In *AAAI Workshop on Artificial Intelligence for Web Search*. AAAI Press, Menlo Park, CA, 35–40.

LARSEN, R. D. AND BOUVIN, N. O. 2004. HyperPeer: Searching for resemblance in a P2P network. In *Proceedings of the 15th ACM Conference on Hypertext and Hypermedia*. ACM Press, 268–269.

LASSILA, O. AND SWICK, R. R. 1999. Resource description framework (RDF) model and syntax specification. W3C Recommendation, World Wide Web Consortium.

LI, J., PEASE, A., AND BARBEE, C. 2002. Experimenting with ASCS semantic search. Project report, Teknowlege Corporation, Palo Alto, CA.

LOWE, D. AND HALL, W. 1999. *Hypermedia and the Web*. John Wiley and Sons, Inc. West Sussex.

LUKKA, T. J. AND FALLENSTEIN, B. 2002. Freenet-like GUIDs for implementing Xanalogical hypertext. In *Proceedings of the 13th ACM Conference on Hypertext and Hypermedia*. ACM Press, 194–195.

MARTIN, D., PAOLUCCI, M., MCILRAITH, S., BURSTEIN, M., MCDERMOTT, D., MCGUINNESS, D., PARSIA, B., PAYNE, T., SABOU, M., SOLANKI, M., SRINIVASAN, N., AND SYCARA, K. 2004. Bringing semantics to Web services: The OWL-S approach. In *Proceedings of the 1st International Workshop on Semantic Web Services and Web Process Composition (SWSWPC 2004)*. Lecture Notes in Computer Science, vol. 3387. Springer, 26–42.

MULLIN, J. 1990. Optimal semijoins for distributed database systems. *IEEE Trans. Softw. Eng. 16*, 5, 558–560.

PEARL, A. 1989. Sun's link service: A protocol for open linking. In *Proceedings of the 2nd Annual ACM Conference on Hypertext*. ACM Press, 137–146.

RATNASAMY, S., FRANCIS, P., HANDLEY, M., KARP, R., AND SCHENKER, S. 2001. A scalable content-addressable network. In *Proceedings of the ACM SIGCOMM Conference*. ACM Press, 161–172.

ROSENBLATT, M. 1956. Remarks on some nonparametric estimates of a density function. *Anna. Mathem. Statist. 27*, 832–837.

ROWSTRON, A. I. T. AND DRUSCHEL, P. 2001. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In *Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms (Middleware'01)*. Springer-Verlag, 329–350.

SILVERMAN, B. W. 1986. Density estimation for statistics and data analysis. In *Monographs on Statistics and Applied Probability*.

SMITH, M. K., WELTY, C., AND MCGUINNESS, D. L. 2004. Owl Web ontology language guide. W3c recommendation, World Wide Web Consortium.

SRIPANIDKULCHAI, K. 2001. The popularity of Gnutella queries and its implications on scalability. Featured on O'Reilly's www.openp2p.com website.

STOICA, I., MORRIS, R., KARGER, D., KAASHOEK, M. F., AND BALAKRISHNAN, H. 2001. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proceedings of the ACM SIGCOMM Conference*. ACM Press, 149–160.

THE OPEN GROUP. 1997. DCE 1.1: Remote Procedure Call. Technical Standard C706, The Open Group.

WIIL, U. K. 1997. Open hypermedia: Systems, interoperability and standards. *J. Digital Inform. 1*, 2.

WIIL, U. K. AND LEGGETT, J. J. 1997. Workspaces: The HyperDisco approach to Internet distribution. In *Proceedings of the 8th ACM Conference on Hypertext*. ACM Press, 13–23.

ZHOU, J. 2004. DDLS: Extending open hypermedia systems into peer-to-peer environments. Ph.D. thesis, University of Southampton.

ZHOU, J., DIALANI, V. K., DE ROURE, D. C., AND HALL, W. 2003. A Distance-based semantic search algorithm for peer-to-peer open hypermedia systems. In *Proceedings of the International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT'03)*. IEEE Computer Society, 7–11.

ZHOU, J., HALL, W., AND DE ROURE, D. 2004. When open hypermedia meets peer-to-peer computing. In *Proceedings of the 15th ACM Conference on Hypertext and Hypermedia*. ACM Press, 266–267.

ZIPF, G. K. 1949. *Human Behavior and the Principle of Least Effort*. Addison-Wesley, Cambridge, MA.

Received June 2005; revised March 2006; accepted July 2006