Diverse Peer Selection in Collaborative Web Search

Le-Shin Wu
School of Informatics
Indiana University
Bloomington, IN 47405, USA
lewu@indiana.edu

Filippo Menczer
School of Informatics
Indiana University
Bloomington, IN 47405, USA
fil@indiana.edu

ABSTRACT

Effective peer selection for intelligent query routing is a challenge in collaborative peer-based Web search systems, especially unstructured networks that do not have any centralized control of peer document collections. In particular, routing a query to multiple peers that provide the same results is a waste of resources. To deal with overlapping document collections we propose a *diverse peer selection* approach for adaptive query routing. This approach takes into account not only which neighbors are the best resource providers for a given query, but also which combinations of neighbors can provide the least redundant results. We validate the feasibility of our proposed algorithm by presenting several simulation experiments conducted with different configurations of peer network environments. Two novel evaluation measures, *distributed precision* and *distributed recall*, are also introduced to provide an effective comparison of different peer network systems. These two performance measures extend the well known IR measures of precision and recall by integrating network costs, namely bandwidth and latency. Our algorithm finds results of equivalent quality using less time and generating less traffic in the presence of varying amounts of document duplication.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; H.3.4 [**Information Storage and Retrieval**]: Systems and Software—*Distributed systems, information networks, performance evaluation (efficiency and effectiveness)*

Keywords

Peer collaborative search, peer selection, overlap, coverage, distributed precision and recall.

1. INTRODUCTION AND BACKGROUND

A growing Web information retrieval literature (e.g., [2, 8, 11, 13]) suggests that the scalability limitations of centralized search engines can be overcome via distributed systems. Peer networks are increasingly seen as a candidate framework for distributed Web search applications.

Peer networks use the social network as the basis for information retrieval and can be classified into different models based on the type of network management. For instance, centralized models maintain a centralized search registry for query routing [2]. Unfortunately, the central control in this approach makes it difficult to adapt the search process to the heterogeneous and dynamic contexts of the peer users. Another type is a decentralized model (as in early versions of Gnutella), in which queries are sent and forwarded blindly by each peer. The problems of this approach are that peers flooded by requests cannot manage the ensuing traffic, and that the topology is uncorrelated with the interests of the peer users. Combined approaches between the above two are also available such as distributed routing tables [13, 14] and hierarchical routing between hub and leaf peers [11]. An alternative is the adaptive model [1, 16], which uses the idea of learning content profiles of neighboring nodes without assuming the presence of special directory hubs.

Peer selection is one of the central components allowing unstructured peer networks to achieve good search performance. When peers issue a query, in addition to matching their own local search engines, the query is forwarded to other peers specializing in that given query by using criteria that measure the degree of similarity between the query and the other peers' knowledge. For example, Crespo and Garcia-Molina [5] use a taxonomy to classify peers and queries into one or more leaf concepts according to their content. Then peers with semantically similar content are grouped together. A query is sent to groups that have higher probability to answer it and propagates only inside those groups. Haase *et al.* [6] use peers' expertise as the basis for peer selection. Peers define their expertise based on a predefined common ontology. Peers whose expertise is similar to the subject of a query are selected as candidates for query routing. Tempich *et al.* [15] use RDF statements as the representation of remote peers' knowledge. To search for a query, a certain number of peers who have knowledge that matches the query, and who have provided quality results for similar queries before, are selected as targets for the query.

However, these existing peer selection algorithms take into account only the predicated query-specific relevance quality of all known peers for peer ranking. Although they can attain good results by greedily maximizing the match between peers and queries (i.e. the precision of each selected peer), they may suffer from suboptimal coverage and a resulting loss in both recall and efficiency. In an extreme case, for example, there may be two peers with identical document collections (which fully overlap). For a traditional peer selection approach, if one peer is selected as a good neighbor, the other peer will definitely be selected as a good neighbor, too. However, forwarding a query to both peers will not return more quality results than submitting to one peer alone.

SAC'09 March 8-12, 2009, Honolulu, Hawaii, U.S.A.

Bender *et al.* [3] use a *Bloom filter* [4] as a tool to account for collection overlap in the course of peer selection. However, this approach necessitates the creation of one Bloom filter for each $(peer, term)$ pair and then the propagation of these Bloom filters across the network. Thus it is not scalable for a large network because of the communication congestion and information storage problems. Another solution computes the collection overlap based on the intersection between the bag union of the results from different collections [7]. But this method needs to establish representatives of each text collection by sending out probing queries and this could be a rather expensive task. In the face of these limitations, we propose a peer selection algorithm that only uses local information, such as queries and responses, as a basis to consider the collection overlap among peers for query routing; therefore peers do not need to make any extra effort to advertise their index information on a global level. The key idea of our proposed algorithm is that the search performance problem due to the overlap of peers' collections can be alleviated by selecting a combination of neighbors which can provide the least redundant results.

2. DIVERSE PEER SELECTION

2.1 Collection Overlap

Conceptually, the collection overlap between two peers corresponding to a particular query can be defined as the ratio of relevant documents present in one peer's collection that also present in the other peer's collection. More specifically, assume p_1 and p_2 are two peers in a peer network. Let $H(p_1, Q)$ be a retrieved relevant set corresponding to a query Q and returned by p_1, and $H(p_2, Q)$ be a retrieved relevant set corresponding to the same query Q and returned by p_2. The collection overlap of p_1 with respect to p_2 corresponding to Q can be calculated as the ratio of the number of identical elements (based on the URL comparison) in $H(p_1, Q)$ and $H(p_2, Q)$ to the number of elements in $H(p_1, Q)$:

$$O(p_1, p_2)_Q = \frac{|H(p_1, Q) \bigcap H(p_2, Q)|}{|H(p_1, Q)|}. \quad (1)$$

Note that the collection overlap between two peers defined in Equation 1 is a non-symmetric measurement, i.e., the value of $O(p_1, p_2)_Q$ is not necessarily equal to the value of $O(p_2, p_1)_Q$, unless the two retrieved sets contain the same number of results. With the overlap measure of Equation 1, we can infer that if the value of $O(p_1, p_2)_Q$ is high and p_2 is already selected as one information provider, then selecting p_1 as another information provider would provide little additional information. By contrast, if the value of $O(p_1, p_2)_Q$ is low and p_2 is already selected as one information provider, then selecting p_1 as another information provider will likely provide us with useful new information.

In addition, the collection overlap measure of Equation 1 does not tell us about the exact overlap between the document collections kept by two different peers. Instead, it provides relative information about the approximate degree of redundancy between two retrieved sets focused on a particular query. However, in a collaborative peer network environment, precise information about the collection overlap between two peers is not necessary because the search scope is limited by the information needs and interests of the peers, and the retrieval algorithms of the local search engines may be different for different peers. Therefore, Equation 1 is sufficient to give us a flavor as to how much more novelty a peer can possibly provide for a particular query if another peer is already selected.

To support adaptive query routing, each peer needs to store and update the collection overlap information between peers with a view to the potential intersection between their returned documents for answering prospective queries. In our design, we let each peer maintain an information matrix W^o for storing collection overlap knowledge. In this matrix, each row corresponds to a pair of peers and each column corresponds to a keyword. Thus an element $w^o_{(p_i,p_j),k}$ of W^o represents the degree of collection overlap of peer p_i to p_j corresponding to keyword k. Note that a peer only stores the overlap information of other peers with whom it interacts. The space required to store W^o and the time to update its entries can be bound by the peer application's available memory or storage.

Algorithm 1 Overlap Profile Update Algorithm

Input: a set of neighbors P, current time t, current query Q

for all $(p_i, p_j) \in P \times P$ and $i \neq j$ **do**
  for all $k \in Q$ **do**
    if $w^o_{(p_i,p_j),k}$ not exist **then**
      $w^o_{(p_i,p_j),k}(t) = a \cdot O(p_i, p_j)_Q$
    else
      $w^o_{(p_i,p_j),k}(t) = (1-a) \cdot w^o_{(p_i,p_j),k}(t-1) + a \cdot O(p_i, p_j)_Q$
    end if
  end for
end for

Algorithm 1 describes how peers update their collection overlap matrix. When a peer receives responses to a query Q, it computes the collection overlap of p_i to p_j corresponding to Q. Based on this calculation, the peer assesses the collection overlap of p_i to p_j with respect to each keyword $k \in Q$ and then updates the element $w^o_{(p_i,p_j),k}$ with a learning rate parameter ($0 < a < 1$). Overlap scores are continuously updated as peers learn about each other, so that it is possible to keep track of dynamic changes in peer content.

2.2 Collection Coverage

In a peer network, the query-specific collection coverage of a peer p can be simply defined as the ratio of the relevant information to a query that is retrieved from p compared to the total relevant information retrieved from the network. But quantifying the amount of retrieved relevant information in order to compute the collection coverage is a rather difficult task. To cope with this problem, we propose a collection coverage approximation (described below) that takes into account not only the size of the peers' responses but also the relevance of each individual result.

The first step for computing the collection coverage is to re-rank the whole set of retrieved documents. A method introduced by Lee [10] is adopted to change the rank of each retrieved document by a peer p into a normalized rank score: $RS_d = 1 - \frac{R_d - 1}{|H_p|}$ where RS_d is the rank score of document d, $|H_p|$ is the number of documents retrieved by p, and R_d is the rank of document d in H_p. (The dependence on a query Q is implicit in this and the following equations to simplify the notation.) Note that a document with lower rank value (more relevant) will yield a higher rank score. By converting ranks into rank scores, the rank scores of documents retrieved are redistributed between 0 and 1 while preserving the original rank relationship among documents. Next, we combine the rank scores of documents returned by different peers by applying a data fusion approach proposed by Shaw and Fox [12] (alternatives may be explored in future work): $CRS_d = \sum_{p \in P} RS^p_d$ where CRS_d is the combined relevance score for document d, RS^p_d is the normalized rank score for document d returned by peer p, and P is the set of peers that respond to the given query. The idea is that a document will be given more ranking weight if more peers return

Algorithm 2 Coverage Profile Update Algorithm

Input: a set of neighbors P, current time t, current query Q

for all $p_i \in P$ **do**
  for all $k \in Q$ **do**
    if $w^c_{p_i,k}$ not exist **then**
      $w^c_{p_i,k}(t) = b \cdot T(p_i, Q)$
    else
      $w^c_{p_i,k}(t) = (1-b) \cdot w^c_{p_i,k}(t-1) + b \cdot T(p_i, Q)$
    end if
  end for
end for

this document with good rank.

Once the combined relevance scores for each retrieved document are in hand, the retrieved documents are re-ranked by these scores and the information score of each individual document can thus be computed as: $IS_d = n - R'_d + 1$ where IS_d is the information score of document d, n is the total number of documents retrieved from the network and R'_d is the new rank of document d according to CRS. Note that, by using IS, a document with lower (better) rank will yield a higher information score.

Finally, the collection coverage of a peer p with respect to a query Q can be computed as the ratio between the sum of the information scores of documents retrieved by p and the sum of the information scores of all documents retrieved from the network:

$$T(p, Q) = \frac{\sum_{d=1}^{n(p)} IS_d}{\sum_{h=1}^{n} IS_h} \quad (2)$$

where $n(p)$ is the number of documents retrieved by peer p for query Q, and n is the total number of documents retrieved from the network for the same query.

In an analogous way to collection overlap, each peer will maintain an information matrix W^c for storing collection coverage knowledge. In this matrix, each row corresponds to a peer and each column corresponds to a keyword. Thus an element $w^c_{p,k}$ of W^c is the collection coverage of peer p estimated for keyword k.

Algorithm 2 describes how peers store and update their collection coverage information. When a peer receives responses to a query Q, it computes the collection coverage of p corresponding to Q. Based on this calculation, the peer assesses the collection coverage of p with respect to each keyword $k \in Q$ and then updates the element $w^c_{p,k}$ with a learning rate parameter ($0 < b < 1$).

2.3 Peer Selection

Given the collection overlap and coverage matrices W^o and W^c, we can now select the actual set of N neighbors among the known peers to whom a query is to be sent. N is a peer parameter that can be set based on available bandwidth and computing power. To this end we also need a system dependent way to assess the similarity between a query and a peer. Wu *et al.* [16] proposed a peer profiling algorithm based on a keyword vector representation of peers. Therefore let us assume here that a similarity function $\sigma(p, k)$ to predict the match of keywords with other peers' knowledge is available based on such a representation.

Algorithm 3 demonstrates how collection overlap and coverage can be used in conjunction with the peer profile representation [16] to select a set of neighbors with potentially higher precision and recall compared to selecting neighbors using the profile-driven similarity function σ alone. The peer selection process begins with selecting a single known peer as a starting point. The first peer selected is the top-ranked peer based only on the query-profile similarity $\sigma(p, Q)$.

Algorithm 3 Peer Selection Algorithm

Input: a set of known peers P, a query Q, number of selected neighbors N, matrices W^o and W^c, similarity function σ
Output: a set of selected peers S ($|S| = N$)

$p_j = argmax_{p_j \in P}[\sigma(p_j, Q)]$
$S \leftarrow \{p_j\}$
$P \leftarrow (P - \{p_j\})$
for $i \leftarrow 2, N$ **do**
  for all $p_x \in P$ **do**
    for all $k \in Q$ **do**
      $C[p_x, k] = (1 - \sum_{p_y \in S} w^o_{(p_x,p_y),k}) \cdot w^c_{(p_x,k)}$
    end for
  end for
  $p_x = argmax_{p_x \in P}[\sum_{k \in Q} \frac{2 \cdot \sigma(p_x,k) \cdot C[p_x,k]}{\sigma(p_x,k) + C[p_x,k]}]$
  $S \leftarrow S \cup \{p_x\}$
  $P \leftarrow (P - \{p_x\})$
end for
return S

Once the first peer is identified, it is added to the list S. The next step is to continue a selection loop until the rest $N - 1$ peers are chosen. In each iteration, we initially compute the similarity σ between each p_x in P and each k in Q. Then we compute the contribution C of each p_x corresponding to each k with respect to S (peers that are already selected as neighbors) by using the collection overlap and coverage information. For example, to compute the contribution C of a peer, say p_2, corresponding to a given keyword, we discount the collection coverage of p_2 by the complement of the collection overlap of p_2 to another peer, say p_1, given that p_1 is already selected for query forwarding. Finally, the peer with the highest value of the harmonic mean (based on the F_1 measure traditionally used in IR to combine precision and recall) of σ and C (summed over all keywords in the query) is selected as another neighbor and moved from P to S. The idea is to send the queries to peers that are likely to have lots of relevant information, but not the same as other selected peers.

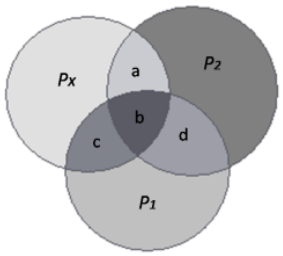

Figure 1: Multi-peer collection overlap. The unique ratio in p_x's collection is $1 - a - b - c$, while its approximation obtained by using pairwise overlap (Algorithm 3) is $1 - \sum_{p_i \in S} w^o_{(p_x,p_i),k} = 1 - [(a+b) + (b+c)] = 1 - a - 2b - c$. Our algorithm underestimates the contribution of p_x by b.

Another challenge needs to be addressed here — the multi-peer collection overlap problem. Figure 1 shows a diagram of the collection overlap relationship with respect to a keyword k among three peers p_x, p_1, and p_2 where p_1 and p_2 are already selected as two content providers. To compute the contribution $C[p_x, k]$ in Algorithm 3, we need to calculate the coverage discount factor corresponding to the collection overlap among p_x, p_1 and p_2. It is not

difficult to observe that the actual value $(1 - a - b - c)$ is larger than the computed value $(1 - a - 2b - c)$. Since our goal is to compare quality responses among known peers for query routing, we hypothesize that using only the pairwise collection overlap information should provide us with an efficient and sufficiently accurate approximation.

3. RESULTS

3.1 Experimental Setup

To illustrate and validate the *diverse peer selection* algorithm described in the previous section we use `sixearch.org`, a freely available intelligent multi-agent Web search application, as an example of peer-based Web search system. A detailed description of its protocols and algorithms is out of the scope of this paper and can be found elsewhere [16].

Two collection testbeds *ASISWOR* and *ASISWR* [9] built upon TREC's WT10g collection were used for our computer simulations. Briefly, to construct *ASISWOR*, documents in TREC's WT10g collection are clustered into different groups based on Web domains. To construct *ASISWR*, the *ASISWOR* was modified by pulling into each document group all the documents from other domains linked by any document in the group. The average document replication rate of *ASISWR* is about 1.2. In addition, five other collection testbeds were also created by duplicating each document group in *ASISWOR* for 3, 5, 10, 15, and 32 times with the purpose of modeling different degrees of document replication within the collection.

The document groups of the collection testbeds are randomly and evenly assigned to 500 synthetic peers. Peers were allowed to discover all other 499 peers and to contact 5 neighbors for any query. The peer network was initialized as a random *Erdos-Renyi* graph, i.e., each peer was assigned 5 random neighbors drawn from a uniform distribution. Each peer in our experiments has 10 queries, randomly selected from the TREC's WT10g data collection, as its own local queries. Finally we set the `TTL` (time to live), a standard technique to limit congestion and loops in any network protocol, to 2 and ran the simulator for about 600 time steps.

Our simulation programs took a snapshot of the network at every time step. In a single time step of the simulator, all of the peers process all of their buffered incoming messages and send all of their buffered outgoing messages.

3.2 Evaluation Measures

Traditional precision and recall measures for evaluating search performance cannot capture the proper costs of distributed search applications because they do not take into account the fact that users are not willing to wait for a long time (latency) and the fact that users have finite bandwidth.

In a distributed information retrieval system, network latency and bandwidth are reflected in how much time the system needs for retrieving documents for a given query, and how many peers it can communicate with. To compare different distributed systems, we have to consider these factors as an integral part of the evaluation, because it is of little use to aim for perfect results if it takes hours to retrieve them. To this end, let us introduce two novel criteria for evaluating distributed search performance: *distributed precision* and *distributed recall*.

Let H be a set of retrieved documents from a network for some query, U a subset of H ($|U| < |H|$) containing only the *unique and relevant* documents, R the set of all relevant documents within the network, and $f(u)$ the maximum routing distance (number of hops) between the originator of the query and the peer(s) returning document $u \in U$. Distributed precision Π is defined as the ratio

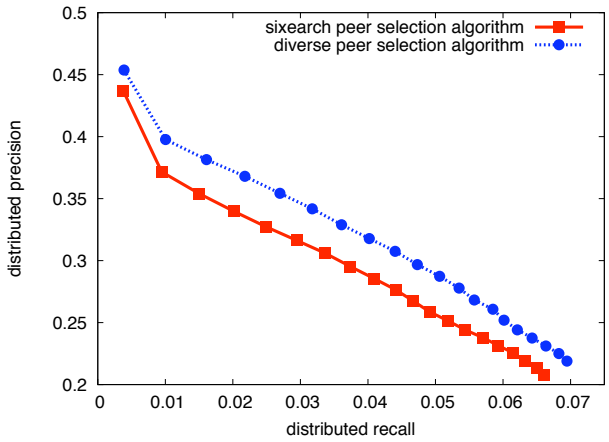

Figure 2: Precision-recall plot for diverse peer selection algorithm and `sixearch.org` peer selection algorithm with 15 duplicates per document. The ranking of the results obtained by each peer is performed after all results are received from other peers and combined.

of the weighted relevant hits count to the total network bandwidth needs (i.e. the total number of retrieved documents, whether unique or duplicate):

$$\Pi = \frac{\sum_{u \in U} f(u)^{-1}}{|H|}. \quad (3)$$

Distributed recall Θ is defined as the ratio of the weighted relevant hits count to the total number of relevant documents within the entire peer network:

$$\Theta = \frac{\sum_{u \in U} f(u)^{-1}}{|R|}. \quad (4)$$

Note that, the numerator in Equations 3 and 4 is used as a weight to capture both relevance and latency. The idea is that the longer the query travel distance, the longer the waiting time. Hence, unlike in traditional information retrieval where every hit counts as the same credit, in the distributed case a hit is given higher credit if it is retrieved from a closer neighbor (i.e. in a shorter waiting time).

Another issue addressed by distributed precision is that we use the total number of retrieved documents $|H|$ (including duplicates) in the denominator of Equation 3 instead of the number of unique retrieved documents. To search in a peer network, the query originators have to download the results provided by other peers to their local disks for further ranking and combining. Therefore, it is inevitable for a peer to spend some bandwidth retrieving some hits that are already known. This will incur transfer costs without yielding benefit for the peer, and distributed precision takes these costs into account when evaluating search performance.

3.3 Experimental Results

Let us analyze the results obtained from our simulations. First we focus on the improvement in search performance between different peer selection algorithms. We also want to see how the search quality changes over different peer network configurations modeled by the degree of document duplication.

Figure 2 shows a precision-recall plot in one simulation with 15 duplicates per document. Diverse peer selection outperforms the `sixearch.org` algorithm by improving both distributed precision and distributed recall. Note that traditional precision and recall would not be able to discriminate between the two algorithms,

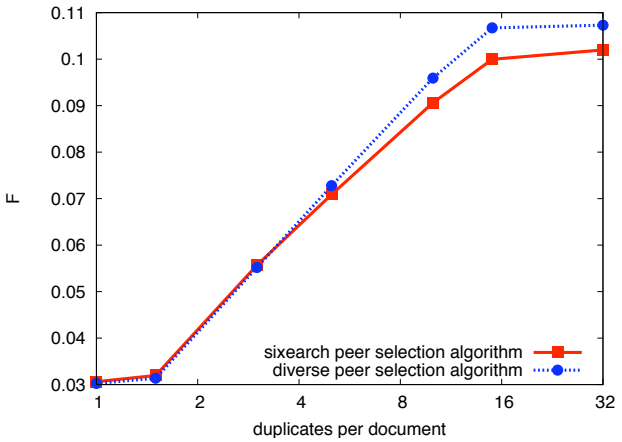

Figure 3: F-measure for diverse peer selection algorithm and `sixearch.org` peer selection algorithm with different degree of document replication within the collection.

because both can eventually find the same number of relevant documents, although in one case we have to wait longer and waste bandwidth by retrieving many copies of the same document. Selecting diverse peers alleviates both of these costs.

To illustrate how the actual search performance changes with the degree of document duplication across peers, we plot the F measure, which combines (distributed) precision and recall through their harmonic mean, versus document duplication for both `sixearch.org` and the diverse peer selection algorithms, as shown in Figure 3. It is not surprising that the search performance of the `sixearch.org` and diverse peer selection both increase with the number of duplicates per document, since there are more documents served by each peer. More meaningful is the favorable comparison between the `sixearch.org` and diverse peer selection algorithms. The latter outperforms `sixearch.org` for higher duplication, and we attribute this to the better cumulative coverage achieved by the diverse peer selection.

4. DISCUSSION

In this paper we introduced a diverse peer selection approach to explore the idea that the coverage limitations of peer-based distributed information retrieval systems can be overcome by integrating collection overlap awareness into the query routing strategy. We also described two novel performance measures, distributed precision and distributed recall, to incorporate the key bandwidth and latency costs of networks into the evaluation of distributed information retrieval systems. Our experiments suggest that diverse peer selection outperforms the `sixearch.org` selection algorithm, an existing query routing approach that does not consider collection overlap and thus cannot select peers for better coverage of the search space.

Several issues are under investigation. We will use parameter sweep experiments to analyze the sensitivity of search performance to network traffic, e.g. due to different numbers of contact neighbors corresponding to various `TTL` values. We also would like to explore different strategies for choosing the first peer in the diverse peer selection algorithm. This could make a significant difference for the search results. The function used to combine the query-profile similarity and the query specific contribution in Algorithm 3 can be further studied. Additionally, we plan to study the use of reinforcement learning algorithms for identifying good neighbors (neighbors that can provide relevant and novel results) not only with their individual performance but also that of their neighborhoods. Finally, the issue of how duplicates can be identified among retrieved documents needs further study. In our current design, we use URL comparisons as a simple way to identify duplicate pages.

5. REFERENCES

[1] R. Akavipat, L.-S. Wu, F. Menczer, and A. G. Maguitman. Emerging semantic communities in peer Web search. In *Proc. of CIKM P2PIR*, 2006.

[2] M. Bawa, R. Bayardo Jr, S. Rajagoplan, and E. Shekita. Make it fresh, make it quick — searching a network of personal webservers. In *Proc. of the 12th Intl. WWW Conf.*, 2003.

[3] M. Bender, S. Michel, P. Triantafillou, G. Weikum, and C. Zimmer. Improving collection selection with overlap awareness in p2p search engines. In *Proc. of the 28th Intl. SIGIR Conf.*, pages 67–74, New York, USA, 2005.

[4] B. H. Bloom. Space/time trade-offs in hash coding with allowable errors. *Commun. ACM*, 13(7):422–426, 1970.

[5] A. Crespo and H. Garcia-Molina. Semantic overlay networks for P2P systems. Technical report, Computer Science Department, Stanford University, 2002.

[6] P. Haase, R. Siebes, and F. van Harmelen. Peer selection in peer-to-peer networks with semantic topologies. In *ICSNW*, pages 108–125, 2004.

[7] T. Hernandez and S. Kambhampati. Improving text collection selection with coverage and overlap statistics. In *Proc. of the 14th Intl. WWW Conf.*, pages 1128–1129, New York, USA, 2005.

[8] S. Joseph. Neurogrid: Semantically routing queries in Peer-to-Peer networks. In *Proc. of Intl. Workshop on Peer-to-Peer Computing*, 2002.

[9] I. A. Klampanos, V. Poznański, J. M. Jose, and P. Dickman. A suite of testbeds for the realistic evaluation of peer-to-peer information retrieval systems. *Lecture Notes in Computer Science*, 3408:38–51, 2005.

[10] J. H. Lee. Analyses of multiple evidence combination. In *Proc. of the 20th Intl. SIGIR Conf.*, pages 267–276, New York, USA, 1997.

[11] J. Lu and J. Callan. Content-based retrieval in hybrid peer-to-peer networks. In *Proc. of the 12th Intl. CIKM Conf.*, 2003.

[12] J. A. Shaw and E. A. Fox. Combination of multiple searches. In *Text REtrieval Conference*, pages 0–, 1994.

[13] C. Suel, T amd Mathur, J.-W. Wu, J. Zhang, A. Delis, M. Kharrazi, X. Long, and K. Shanmugasundaram. ODISSEA: A Peer-to-Peer architecture for scalable Web search and information retrieval. In *Intl. Workshop on the Web and Databases (WebDB)*, 2003.

[14] C. Tang, Z. Xu, and S. Dwarkadas. Peer-to-peer information retrieval using self-organizing semantic overlay networks. In *Proc. of SIGCOMM '03*, 2003.

[15] C. Tempich, S. Staab, and A. Wranik. REMINDIN': Semantic query routing in peer-to-peer networks based on social metaphors. In *Proc. of the13th Intl. WWW Conf.*, pages 640–649. ACM Press, 2004.

[16] L.-S. Wu, R. Akavipat, and F. Menczer. 6S: Distributing crawling and searching across Web peers. In *Proc. of the IASTED Intl. Conf. on Web technologies, Applications, and Services*, Calgary, Canada, 2005.