# Exploiting the Properties of Query Workload and File Name Distributions to Improve P2P Synopsis-based Searches

William Acosta Surendar Chandra
Department of Computer Science and Engineering
University of Notre Dame
$\{wacosta, surendar\}@nd.edu$

***Abstract*—Modern P2P systems use hybrid searches to improve search efficiency. They use a synopsis of neighborhood content to determine whether to use a structured or unstructured overlay to satisfy a particular query. Because of their size restrictions, a synopsis cannot hold all the terms from every file in the neighborhood. The challenge is to choose the terms that should be represented in the synopsis. In this work, we investigated the distribution of query terms and file terms in Gnutella networks. We observed that there was a mismatch between terms that were popular among file names and the terms that were popular among the queries generated by the user. Because the query behavior changed with time, a synopsis based on only static set of popular file terms was ill-suited to support efficient searches. We used these observations to design a synopsis creation algorithm that dynamically adapted to the query workload and selected terms for the synopsis to reflect popular terms in both the query workload and file distribution. Our preliminary experimental analysis showed that our Query-Adaptive synopsis improved the search performance over the traditional file-based synopsis model.**

## I. Introduction

Searches in P2P networks can be generally divided into two categories: informed and uninformed. Informed searches make use of information about the content at neighboring peers to make routing decisions for queries. In contrast, uninformed searches, like flooding and random walks, do not use information about neighboring peer content to make routing decisions.

As P2P networks grow [1] and become more complex, the limitations of uninformed searches affect performance of the entire system [2]. The trend for improving P2P search performance is, therefore, toward using informed search mechanisms. For example, several indexing schemes [3], [4] have been proposed to improve search performance. More recently, hybrid approaches [5], [6] have been developed to take advantage of the benefits of both structured and unstructured searches. These hybrid P2P systems use information about the shared content to make query routing decisions.

Informed searches require that information about the content at each peer be summarized in a *synopsis* and shared to other peers. Prior work had focused on the data structures used to represent the synopsis [7], [4] and the algorithms used to distribute and maintain the synopsis [5]. In this paper, we examined the problem of selecting appropriate content for the synopsis. Because a peer's synopsis needs to be distributed to many other peers, there is incentive to minimize the size of the synopsis. However, as the synopsis becomes constrained, the contents of the synopsis must be carefully chosen to maximize the ability of the synopsis to resolve queries. Our prior work [8], [9] showed that there are few terms in common ($< 20\%$) between the popular query terms and the popular file terms. Therefore, a synopsis constructed using only the static set of popular file terms is ill-suited to support efficient searches.

In this paper, we show that temporal patterns in popular query terms can be exploited to construct an effective synopsis. We show that the set of overall popular terms remained stable over time. However, we show that some terms that were not popular overall, became popular for a period of time. Therefore, a synopsis must consider the relationship between query terms and file terms, it must also consider both the overall popular and the transiently popular query terms. We propose a model for selecting synopsis terms that adapts to the query workload. We periodically updated the synopsis to reflect observed trends in popular query terms and their relationship to a peer's shared files. Our Query-Adaptive synopsis monitored the queries routed through a peer and tracked the intersection of popular query terms and the terms in the file annotations. At periodic intervals, the synopsis was updated to reflect the set of terms that occurred both in the file annotations and in the monitored query trace. We varied the size of the synopsis and the evaluation interval for updating the synopsis. Our results showed that our Query-Adaptive synopsis improved the synposis hit rates from 30% to about 60% over the file term only synopsis for a synopsis that could fit into a 1,500 byte packet (1,250 terms in a Bloom filter representation). Additionally, the increase in performance resulted in only 3% false positive rate.

The rest of this paper is organized as follows. In Section II we describe the intuition behind our Query-Adaptive synopsis model and explore the characteristics of a query workload captured from a live system. We then describe our Query-Adaptive synopsis creation algorithm in Section III and provide an analysis of its performance in Section IV. We then place our work in the context of related research in Section V and provide concluding remarks in Section VI.

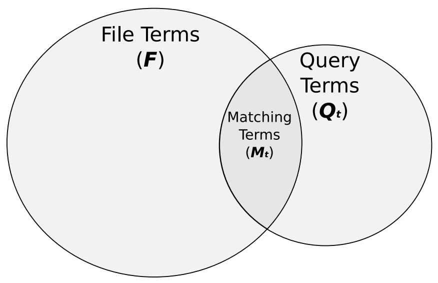
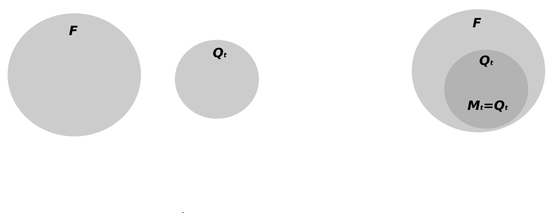


Fig. 1. Relationship between the query terms for a given time interval $t$ and the file terms. Only queries whose terms are in $M_t$ are matched.



(a) $M = \emptyset$

(b) $M_t = Q_t$

Fig. 2. Boundary conditions for $M_t$. If $M_t = \emptyset$ no queries matched. If $M_t = Q_t$, then there is a potential to match all queries. Because $Q_t$ is not static, $M_t$ also varies with time.

## II. Characteristics of the Query Workload

A central component of our design was understanding how query and file term distributions affected the ability to resolve queries. We first focused on examining the relationship between query terms and the file terms of the shared objects. We then analyzed the temporal patterns of query terms to determine which terms should be included in a synopsis.

### *A. Understanding The Query Workload and the Means by Which Queries are Matched*

First we examined how the relationship between query and file terms affects the ability to resolve queries. Given a set of files shared by peers and a query trace, $F$ is the set of terms that appear in annotations of the shared files, and $Q_t$ is the set of terms that appear in the query trace for a given time interval $t$. $F$ is relatively static, but $Q_t$ is expected be vary over time. The shared matching terms $M_t = F \cap Q_t$ is the intersection of file and query terms at a given time (illustrated in Figure 1). If $M_t = \emptyset$, then no queries could be matched because no files exist with terms that matched the terms used in the queries (illustrated in Figure 2(a)). If $Q_t \subseteq F$ then $M - t = Q_t$ thus all queries could potentially be matched (illustrated in Figure 2(b)). Because, $M_t$ depends on $Q_t$, it is important to understand the temporal patterns of $Q_t$ in order to construct synopses that can effectively resolve queries.

In real-world environments, constraints such as bandwidth limitations require that the synopsis size be bounded. An informed search mechanism, therefore, requires that the distributed synopsis describe the peer's content in a manner that maximizes the number of queries that are correctly matched while minimizing the information that is required to describe the peer's content. In the next section we explore the query workload and present preliminary findings with respect to the temporal patterns of popular query terms.

### *B. Persistence of Popular Query Terms*

In this section, we examined the characteristics of a real-world query workload with respect to the temporal patterns of popular query terms. We were interested in determining the appropriate set of terms to include in a synopsis of a peer to facilitate informed search mechanisms. We modified the source code of Phex [10], an open source Gnutella client to capture queries on the network. The client connected to the network and logged every incoming and outgoing query that passed through it. We ran the query capturing client for one week in June, 2007 and captured over 2.5M queries.

We were interested in determining the stability of the set of popular query terms over time. We defined two classes of popular terms: *overall popular* ($Q'$) and *interval popular* ($\hat{Q}'$). The *overall popularity* of a term was computed by counting all prior occurrences of the term in the query workload. The *interval popularity* counted only the occurrences of the term in the current interval.

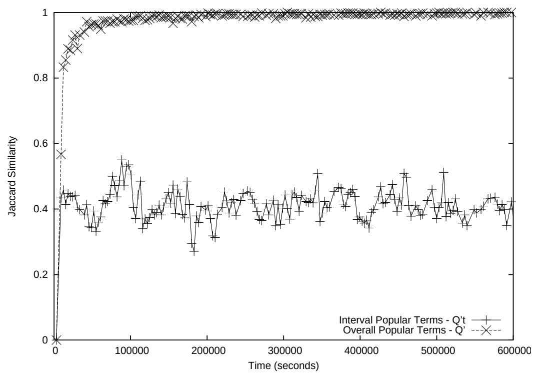


Fig. 3. Jaccard similarity over time for the set of top 1,000 overall ($Q'_t$) and interval popular terms ($\hat{Q}'_t$) vs. the set of overall popular and interval popular terms from the previous evaluation interval, ($Q'_{t-1}$) and ($\hat{Q}'_{t-1}$), respectively. Analysis of a 1 week query trace with the evaluation interval for computing popular terms set to 60 minutes.

We were interested in determining how the sets of both the *overall popular* ($Q'$) and *interval popular* ($\hat{Q}'$) query terms changed between evaluation intervals. We use the Jaccard index to compare the similarity of the set of popular query terms to the previous set of popular query terms. The Jaccard index provides a means to compare the similarity between two sets of objects. The values range between 0 (entirely dissimilar) and 1 (identical). We analyzed a one week query trace using a 60 minute evaluation interval and computed $Jaccard(Q'_t, Q'_{t-1})$ and $Jaccard(\hat{Q}'_t, \hat{Q}'_{t-1})$ at each evaluation interval. Figure 3 plots the Jaccard similarity between the popular query terms for an interval and the terms that were popular in the prior interval. We found that that after a short stabilization time, the set of *popular terms* remained relatively constant exhibiting a high Jaccard similarity value ($> 90\%$). The set of *interval popular* terms varied significantly, but maintained a stable Jaccard similarity value of approximately 40%. These results indicated that many popular terms from one evaluation interval were likely to be popular in the next interval. In the next section we describe an algorithm that exploits these findings to create a synopsis that adapts to the query workload.

## III. Query-Adaptive Synopsis Algorithm

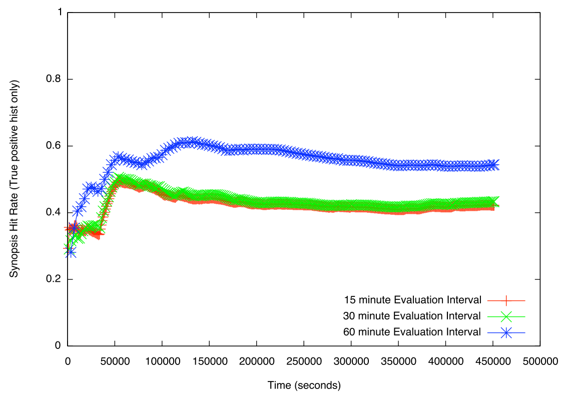
Our Query-Adaptive synopsis exploited the query workload characteristics from the previous section to identify the appropriate terms to include in the synopsis. Specifically, the algorithm leveraged the fact that popular query terms tended to remain popular and gave them higher priority when selecting terms to include in the synopsis. Conversely, because the popular file terms had little similarity to the set of query terms observed by a peer, the algorithms assigned them the lowest priority when selecting terms for the synopsis. First, the algorithm selected the overall popular query terms that matched the file terms during the evaluation interval ($Q'_t \cap F$). Next, it determined the interval popular terms that were also popular in the previous interval ($\hat{Q}''_t = \hat{Q}'_t \cap \hat{Q}'_{t-1}$). It then selected those terms from that set that also matched the file terms ($\hat{Q}''_t \cap F$). In the next sections, we present our analysis of the performance of this Query-Adaptive synopsis algorithm.

## IV. Performance Evaluation of the Query-Adaptive Synopsis

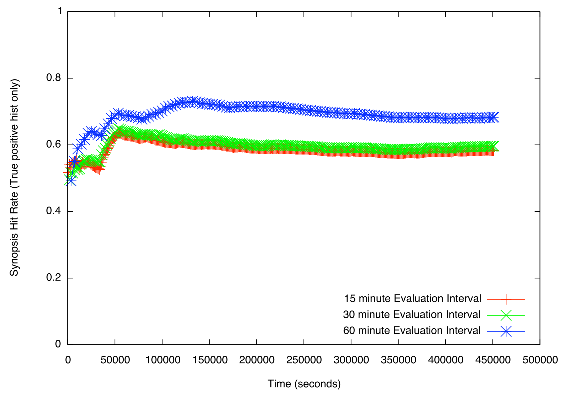
In Gnutella, ultrapeers typically index the content of their leaf peers and respond to queries on their behalf. However, queries between ultrapeers are still largely routed using a TTL-bounded flooding mechanism. For our analysis, we examined the ability of a Query-Adaptive synopsis to effectively represent the content of a set of leaf peers in order to facilitate informed searches such as in a hybrid P2P approach. We crawled the Gnutella network to discover files shared by peers. Our crawl yielded over 20M shared objects from over 38K peers. We then selected a random set of peers from the crawl to serve as leaf peers. We aggregated the set of files shared by these peers to form the set of files that our ultrapeer synopsis would index. From [11] we know that Gnutella ultrapeers typically have 64 leaf peers. On average, a random set of 64 peers from our data set yielded 20,000 unique file terms. We set the synopsis limit to 5000, 2500, and 1250 terms. This represented 25%, 12.5%, and 6.25% of the unique file terms from the leaf peers, respectively. A synopsis of 1250 terms can be represented by a Bloom filter [12] of only 1500 bytes with a 1% false positive rate.

### A. Hit Rate Analysis for a Query-Adaptive Synopsis

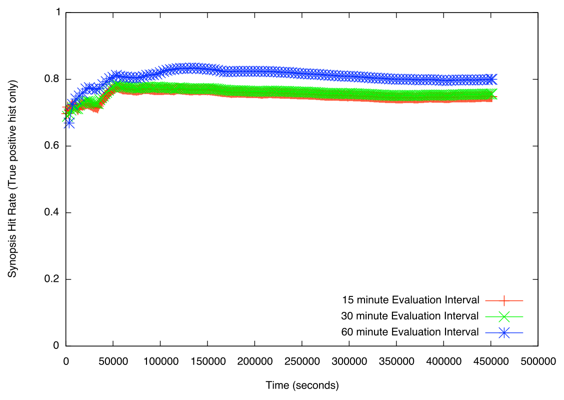
Our aim was to evaluate how well the synopsis described the content of the set of leaf peers by examining the true positive (TP) hit rate of the synopsis. We first evaluated all queries against the full set of crawled peers and objects to create a database of the results for each query. Next, we selected a random set of peers (leaf peers) and created a synopsis with the specified term limit to capture the terms of the files shared by the leaf peer set. We then identified all queries with at least one matching file belonging to a peer in the leaf peer set. We used this subset of queries as the query trace for evaluating the performance of the synopsis. We re-issued this set of queries against the synopsis and tracked how many of those queries were positively matched by the synopsis to determine the true positive hit rate for the synopsis.



(a) 1250 Terms in Synopsis



(b) 2500 Terms in Synopsis

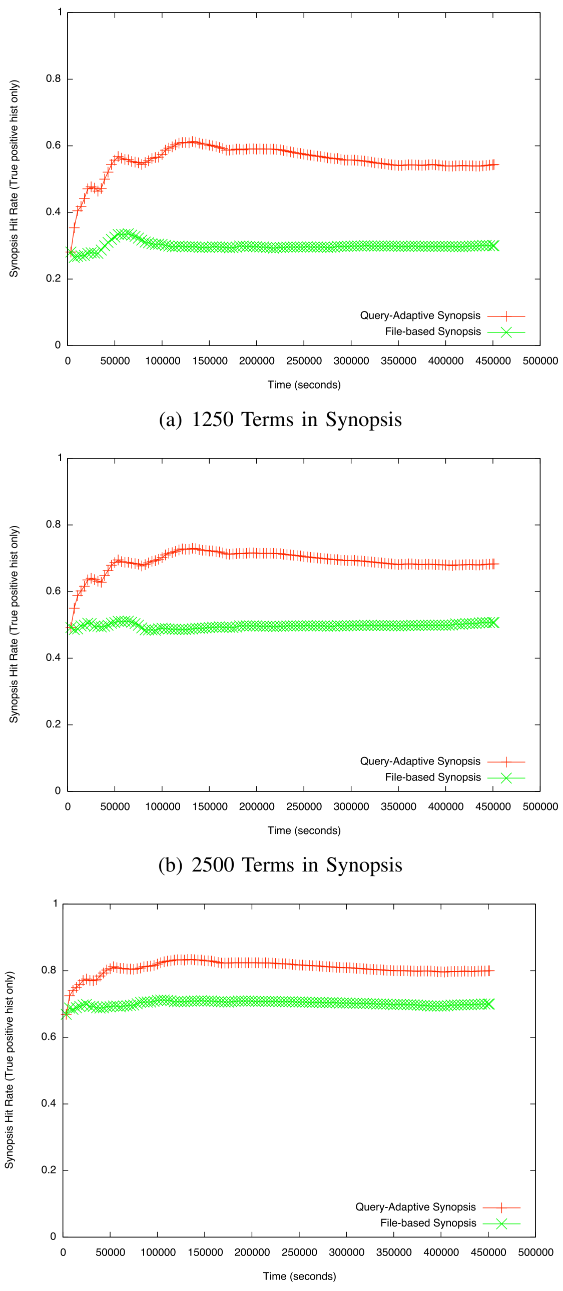


(c) 5000 Terms in Synopsis

Fig. 4. Success rate over time for Query-Adaptive synopsis search using synopsis evaluation intervals of 15, 30, and 60 minutes.

| | 15 min | 30 min | 60 min |
|---|---|---|---|
| 1250 Terms | 2.9% | 2.8% | 2.7% |
| 2500 Terms | 6.7% | 6.6% | 6.8% |
| 5000 Terms | 11.9% | 11.7% | 11.6% |

TABLE I
FALSE POSITIVE RATE FOR DIFFERENT SYNOPSIS TERM LIMITS AND EVALUATION INTERVALS.



(a) 1250 Terms in Synopsis

(b) 2500 Terms in Synopsis

(c) 5000 Terms in Synopsis

Fig. 5. Comparing the success rates of the file-based and Query-Adaptive synopsis searches over time. Evaluation interval for the Query-Adaptive synopsis is 60 minutes.

In Figures 4(a), 4(b), and 4(c) we plot the true positive (TP) hit rate over time of a search based on our Query-Adaptive synopsis. We varied the synopsis evaluation interval from 15 minutes to 1 hour and the synopsis size from 1,250 to 5,000 terms. We observed that with only 25% of the terms from the leaf peer set (5,000 terms), the synopsis achieved an 80% TP hit rate rate. Additionally, the synopsis evaluation interval did not significantly affect the performance of the synopsis when the term limit was 2500 and 5000 terms. However, we observed that even with only 6% of the file terms from the leaf set (1,250 terms), the Query-Adaptive synopsis achieved nearly a 60% TP hit rate when the evaluation interval was 60 minutes. In contrast, the shorter synopsis evaluation intervals (15 and 30 minutes) for a synopsis of 1,250 terms only achieved a 40% TP hit rate rate.

### *B. False Positive Rate Analysis*

Because the synopsis was limited in the number of terms it contains, it could not perfectly capture all the files in the leaf set. The ability of the synopsis to accurately describe the content of the ultrapeer/leaves is, therefore, determined by both the TP hit rate and the false positive (FP) rate. We evaluated all queries from our week-long query trace against a Query-Adaptive synopsis using different synopsis term limits and evaluation intervals to determine the FP rate. The results are summarized in Table I. For all evaluation intervals, the FP rate increased as the synopsis term limit increased. The reason for this is that the synopsis aggregated terms from multiple peers. All terms from a given query may have existed in files from the leaf set, but not necessarily in the same file. As the term limit for the synopsis increased, the probability of this event occurring also increased. Nevertheless, the false positive rate for the synopsis was reasonable (11.9%) even when the term limit was relatively large (5000 terms). When the term limit was 1250 terms, the false positive rate was less than 3%. These suggests that a synopsis with a small term limit and a reasonable evaluation interval (60 minutes) can achieve a good TP hit rate (60%) as well as a low FP rate.

### *C. Performance Comparison Against a Traditional File Term-based Synopsis*

In the previous sections we described the performance of a Query-Adaptive synopsis with respect to the TP hit rate and false positive rate. In this section we compare the performance of a Query-Adaptive synopsis to the traditional synopsis construction model.

Prior synopsis construction approaches only consider the terms in the shared files of the peers [5]. The synopsis is typically constructed by selecting the top terms that occur in the set of files for the peer. An advantage of this approach is that the set of files shared by a given peer remains relatively static over time. Therefore, a peer can construct a synopsis and distribute it without needing to send updates to the recipients of the synopsis.

We examined the relative performance of a file-based synopsis and our Query-Adaptive synopsis. We varied the synopsis term limit for both the file-based and our Query-Adaptive synopsis. Additionally, for our Query-Adaptive synopsis, we used a 60 minute evaluation interval. Figures 5(a), 5(b), and 5(c) plots the true positive (TP) hit rate over time for each synopsis using different term limit settings. For all term limit settings, our Query-Adaptive synopsis outperformed the traditional file-based synopsis. When the term limit was low (1250 terms), our Query-Adaptive synopsis produced a TP hit rate nearly double that of the file-based synopsis. When the term limit was 5000 terms, our synopsis still had a higher TP positive rate, but the performance gap was reduced. In scenarios that require a small synopsis, adapting the content of the synopsis intelligently can significantly improve the performance of search.

## V. Related Work

Previous research has investigated various search techniques in unstructured P2P networks. More recently, however, the trend has been toward hybrid P2P systems [5], [6]. Hybrid P2P systems address the problem of improving search performance by leveraging both structured [13], [14] and unstructured mechanisms to resolve queries. QRank [15] ranks queries by weighting the terms used in the queries and computing a difficulty rank for the query. If the query contains ”easy” terms, QRank uses a simple flooding search to locate the content. However, if the query is ”hard” it uses a DHT search. Computing the difficulty ranking of the terms in the query is central to QRank. The authors distribute statistics about relative popularity of terms that appear in the shared files to all peers in the system. QRank then uses these global statistics to compute the difficulty rank of each term. Our approach also considered the popularity of file terms when creating the synopsis. However, we also actively monitored the queries and adapted the synopsis to include the file terms that matched the terms from the observed query workload. This yielded synopses that described the content at each peer better than the current file-based approaches as shown in our results by improved search performance.

A further development in the hybrid P2P model made use of the synopsis to obtain one-hop search latency for queries matched against the synopsis. For example, in ASAP [7], the content at each peer (synopsis) was proactively distributed to other peers as an advertisement (ad in brief). Peers stored interesting ads from other peers and used the stored ads to make query routing decisions. This achieved a one-hop search latency if the content was discovered in the set of ads stored by the local peer. Rather than specify how the synopsis, or ads, are distributed and used for search, our approach focused on identifying appropriate content to include in the distributed synopsis. Our aim was to exploit the changes in query term distribution with respect to popular queries and their matching file terms. This provided a description of the content at each peer that more closely matched the expected queries.

## VI. Conclusion

In this paper we have presented an analysis of a captured query workload. We showed that popular query terms tend to remain popular and used this observation to create a Query-Adaptive synopsis that could represent the shared content accurately with respect to the query workload. We also showed that synopsis-based search mechanisms that rely only of the popular file terms were ill-suited to resolve searches effectively. The discrepancy between file term and query term popularity yielded a synopsis that did not accurately describe the shared content for a peer with respect to the query workload. We propose that synopsis-based searches, as in some hybrid P2P systems, employ a Query-Adaptive model for constructing the synopsis. Using a Query-Adaptive synopsis allows the hybrid P2P systems to resolve queries matched against the synopsis with one-hop search latency thereby reducing the overall query response time.

## Acknowledgments

This work was supported in part by the U.S. National Science Foundation (CNS-0447671).

## References

[1] A. H. Rasti, D. Stutzbach, and R. Rejaie, “On the long-term evolution of the two-tier gnutella overlay,” in *IEEE Golbal Internet*, 2006.

[2] W. Acosta and S. Chandra, “Trace driven analysis of the long term evolution of gnutella peer-to-peer traffic,” in *Proceedings of the Passive and Active Measurement Conference (PAM'07)*, Louvain-la-Neuve, Belgium, April 2007.

[3] C. Tang and S. Dwarkadas, “Hybrid global-local indexing peer-to-peer information retrieval,” in *Proceedings of the 1st Symposium on Networked Systems Desing and Implementation (NSDI'04)*, 2004.

[4] B. T. Loo, J. M. Hellerstein, R. Huebsch, S. Shenker, and I. Stoica, “Enhancing p2p file-sharing with an internet-scale query processor,” in *Proceedings of the 30th Very Large Data Bases Conference (VLDB'04)*, 2004.

[5] M. Zaharia and S. Keshav, “Gossip-based search selection in hybrid peer-to-peer networks,” in *Proceedings of the 5th International Workshop on Peer-to-Peer Systems (IPTPS'06)*, 2006.

[6] B. T. Loo, R. Huebsch, I. Stoica, and J. M. Hellerstein, “The case for a hybrid p2p search infrastructure,” in *Proceedings of the 3rd International Workshop on Peer-to-Peer Systems (IPTPS'04)*, 2004.

[7] H. Cai, P. Gu, and J. Wang, “Asap: An advertisement-based search algorithm for unstructured peer-to-peer systems,” in *Proceedings of the IEEE International Conference on Parallel Processing (ICPP'07)*, 2007.

[8] W. Acosta and S. Chandra, “Understanding the practical limits of the gnutella p2p system: An analysis of query terms and object name distributions,” in *Proceedings of the ACM/SPIE Multimedia Computing and Networking (MMCN '08)*, San Jose, CA, April 2008.

[9] ——, “On the need for query-centric unstructured peer-to-peer overlays,” in *Proceedings of the Fifth IEEE International Workshop on Hot Topics in Peer-to-Peer Systems*, Miami, FL, April 2008.

[10] “The phex gnutella client,” http://phex.kouk.de.

[11] D. Stutzbach, R. Rejaie, and S. Sen, “Characterizing unstructured overlay topologies in modern p2p file-sharing systems,” *IEEE/ACM Transactions on Networking*, 2007.

[12] B. H. Bloom, “Space/time trade-offs in hash coding with allowable errors,” *Commun. ACM*, vol. 13, no. 7, pp. 422–426, 1970.

[13] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, “Chord: A scalable peer-to-peer lookup service for internet applications,” in *Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*. ACM Press, 2001, pp. 149–160.

[14] A. I. T. Rowstron and P. Druschel, “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 Heidelberg*. Springer-Verlag, 2001, pp. 329–350.

[15] H. Chen, H. Jin, Y. Liu, and L. M. Ni, “Difficulty-aware hybrid search in peer-to-peer networks,” in *Proceedings of the IEEE International Conference on Parallel Processing (ICPP'07)*, 2007.