On the need for query-centric unstructured peer-to-peer overlays

William Acosta and Surendar Chandra
University of Notre Dame, Notre Dame, IN 46556, USA
{wacosta,surendar}@nd.edu

***Abstract*—Hybrid P2P systems rely on the assumption that sufficient objects exist nearby in order to make the unstructured search component efficient. This availability depends on the object annotations as well as on the terms in the queries. Earlier work assumed that the object annotations and query terms follow Zipf-like long-tail distribution. We show that the queries in real systems exhibit more complex temporal behavior. To support our position, first we analyzed the names and annotations of objects that were stored in two popular P2P sharing systems; Gnutella and Apple iTunes. We showed that the names and annotations exhibited a Zipf like long tail distribution. The long tail meant that over 98% of the objects were insufficiently replicated (less than 0.1% of the peers). We also analyzed a query trace of the Gnutella network and identified the popularity distribution of the terms used in the queries. We showed that the set of popular query terms remained stable over time and exhibited a similarity of over 90%. We also showed that despite the Zipf popularity distributions of both query terms and file annotation terms, there was little similarity over time ($< 20\%$) between popular file annotation terms and popular file terms. Prior P2P search performance analysis did not take this mismatch between the query terms and object annotations into account and thus overestimated the system performance. There is a need to develop unstructured P2P systems that are aware of the temporal mismatch of the object and query popularity distributions.**

I. Introduction

A vast body of prior work had investigated various aspects of building and maintaining overlays for managing the stored objects in P2P systems. Structured overlays such as Pastry [1] and Tapestry [2] used a distributed hash table data structure to route queries. Unstructured P2P systems such as Gnutella [3] maintained a distributed overlay for routing requests while using flooding or random walks [4] to locate objects. Hybrid P2P systems [5] used a localized unstructured search before using a global structured lookup to improve the search performance.

Structured P2P systems required an exact object name match in the query string. Unstructured P2P systems used term matching for their queries; the system searched for all objects that matched the set of terms in the query string. The performance of these systems critically depended on the naming and annotation mechanisms that described the objects. However, peers were independent entities. Peers annotated and described objects independently from other peers who queried and requested these objects. It is important to understand how peers actually named and annotated objects as well as how those annotations relate to the queries used to locate the objects.

Many prior P2P systems had made no special assumptions on how the shared objects were actually annotated or how users queried for those objects. Recently, some work observed a Zipf like long tail distribution [6], [7] of object annotation and the query terms. To confirm these observations, first we experimentally analyzed object annotations in two popular systems: Gnutella and iTunes sharing systems. Gnutella named objects using a single name while iTunes used a richer set of annotations. iTunes also used the Gracenote (www.gracenote.com) service to obtain meta-data used to describe objects. Since many P2P systems had been deployed and popular for a number of years, we expected that the naming mechanisms would have converged to a state wherein clients can reasonably expect to search and find the contents of interest. We based our analysis on the object names and annotations alone; we were not concerned with analyzing the semantic equivalence. Note that much of the P2P shared contents were multimedia objects [7]; efforts to analyze the media contents (similar to one used by Pucha et al. [8]) in order to recognize whether two objects were identical will likely fail because the same object could have been transcoded to various formats with imperceptible differences. For example, transcoding a `.wma` file to `.mp3` would appear different at the byte level.

Our analysis showed that the annotations exhibited a Zipf like behavior for analyzing the system in terms of names, as well as annotations such as *artist*, *album* and *genre*. Over 98% of the objects were replicated in less than 0.1% nodes. Next, we analyzed queries from a one week trace from the Gnutella network and tracked the popularity of query terms during this time. We show that the set of popular query terms remained stable over time. However, there was little similarity ($< 20\%$) between popular query terms and popular file annotation terms during each evaluation interval. This suggests that there was a temporal mismatch between terms that were popular among file names and the terms that were popular among the queries generated by the user. We show that prior work had overestimated the ability for unstructured P2P systems to locate popular content. We also showed that hybrid P2P systems will likely perform worse than the corresponding structured P2P systems because of the initial failure of the unstructured search component. Our position in this paper was that P2P systems must consider the temporal relationship between queries and files to support efficient searches. The design of such P2P systems is the subject of our ongoing research. In [9], we present preliminary results of a system

that can exploit the temporal patterns in query term popularity to build adaptive synopses of contents for each peer.

The rest of the paper is organized as follows. Section II describes our experiment setup. We describe our analysis of Gnutella and iTunes object annotations in Section III. We then present our analysis on the relationship between query terms and file annotation terms in Section IV. Section V describes the implications of our findings. Section VI places our work in the context of prior research. We conclude with a general discussion in Section VII.

II. System Architecture

Our experiments were designed to analyze the object annotations and their relationship to the query workload. First we analyzed the object annotations in Gnutella and Apple iTunes. We also analyzed the user queries in the Gnutella network.

A. Collecting Gnutella object annotations and queries

We developed a file crawler similar to *Cruiser* [10] that crawled the Gnutella network and queried each peer for a list of shared objects. The crawler first performed a topology crawl to discover peers connected to the Gnutella network. It then performed a file crawl by connecting to each discovered peer and requesting a list of its shared files. All file names were transmitted from other peers using UTF-8. UTF-8 allowed strings containing both single byte as well as multi-byte character sets. We performed a crawl in October 2006 and discovered 18.6 million objects (7.2 million unique objects) on 41,910 peers. Our crawl in April 2007 discovered over 21 million objects (8.1 million unique objects) shared by 37,572 peers. Gnutella described objects using a single name. Users were free to impose their own structure on this name.

To collect our query trace, we modified the source code of Phex [11], 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 April 2007 and captured over 2.5M queries.

B. Collecting iTunes object annotations

Apple iTunes allowed users to selectively share (using playlists) their song collection with other iTunes clients; this sharing was turned OFF by default. Once the sharing was turned ON, the default behavior was for the clients to share all their local objects. Recent versions of iTunes restricted the sharing to be within the same sub-network. Also, iTunes only allowed sharing objects with five distinct IP addresses within a 24 hour period (even for contents such as podcasts that may allow unrestricted distribution). Users can password protect their shares in order to avoid this five user limitation. Songs purchased from the iTunes store may be protected by Apple Fairplay DRM. The DRM-protected songs cannot be publicly consumed, though the they can be shared. To play these songs, a computer had to be explicitly authorized. iTunes allows users to manage PDF, audio and video objects, though PDF objects cannot be shared with other clients. We collected information about these shared-protected contents in our experiments.

Apple iTunes allowed for a richer annotation with categories such as *Genre*, *Artist* and *Album*. Songs that were purchased from the Apple store embedded these annotations while songs ripped by the user were automatically annotated by the iTunes client using the Gracenote service. End users were allowed to change these annotations.

We modified AppleRecords [12] to connect to iTunes clients and log the songs that were shared by the user. We encountered 620 unique iTunes shares within the University. Of these, 145 shares were password protected, 313 shares returned a busy error code and 111 shares were being blocked by network firewalls. Clients were likely busy because the tracing program violated the 5 unique IPs addresses per 24 hour restriction. We successfully collected the object share information from 239 unique users (significantly smaller than the observed number of peers in Gnutella networks). These users shared 533,768 objects of which 171,068 objects were unique. For each of the shared objects, we logged the following attributes: track id (set by the particular iTunes client), track name, album name, artist name, track number, genre, (the system did not enforce the genres; the users were allowed to create their own genres easily), user song rating, object format (e.g. MP3, AAC audio, MPEG-4 audio book), length of object (in milliseconds), sample rate (e.g. 44100 Hz), song bitrate (in kbps), object size (in bytes), BPM, disk count, disk number, song description, comments, date song was added and the date that the song was last modified. We conducted the experiment during late April/early May in 2006. Since Apple restricts sharing within the same sub-network, our collection agent had to be run within the sub-network. Notre Dame partitions its local network in a number of different VLANs. We monitored one of these dormitory VLANs as well as several wired and wireless VLANs within the campus. During this duration, the entire campus wireless LAN infrastructure was configured to route all Zeroconf service discovery packets to the monitoring station. This allowed us the flexibility of not installing a monitoring station inside each of the campus WLANs. We also collected some traces from another school (which wished to remain anonymous). We did not notice any significant content differences between the users from various networks of the two schools.

III. Analyzing shared object annotations

Next, we describe the results of our analysis of the shared contents that were available in Gnutella and iTunes. Gnutella objects represent P2P systems that used a single name to represent an object. On the other hand, iTunes described objects using a richer set of annotations. Our analysis was agnostic to any particular overlay topology.

A. Object annotations in Gnutella

We describe our analysis of object names that were captured during April 2007. We observed similar results for our October 2006 data set. The trace consisted of over 21 million objects

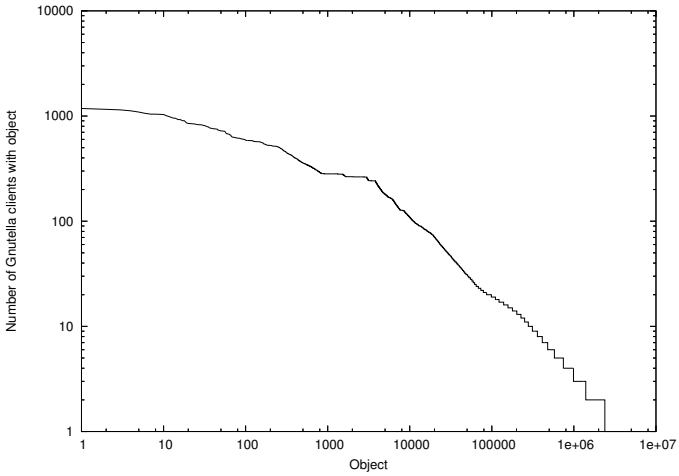


Fig. 1. Number of Gnutella clients with object (April, 2007)

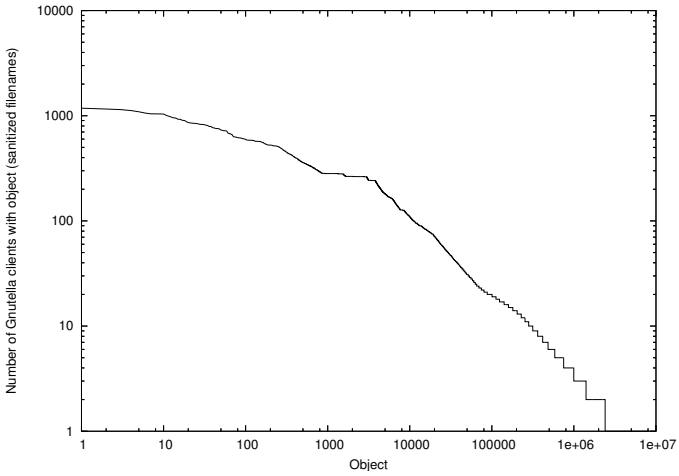


Fig. 2. Number of Gnutella clients with object. Object names were sanitized by removing capitalization and special characters (e.g., dashes) (April, 2007)

(8.1 million unique objects) that were shared by 37,572 peers. For each object in the trace, we plotted the number of replicas in Figure 1. For this analysis, replicas were defined as files with identical names. We also sanitized the file names by removing capitalization and special characters such as dashes and plotted the file name popularity in Figure 2. In the case of sanitized filenames, replicas were defined to be files with identical names after the sanitation process. From Figure 1, we note that about 5.7 million of the 8.1 million unique objects were available in a single peer; about 70.5% of the objects were not replicated in any of the 37,572 peers. About 99.5% of the objects were replicated in less than 0.1% (37) of the peers. Similarly, from Figure 2, we noted that sanitizing the names decreased the number of unique objects to about 7.9 million, 5.5 million of which were available in a single peer; about 69.8% of the objects were not replicated. About 99.4% of the objects were replicated in less than 0.1% (37) of the peers.

We manually browsed these object names and sometimes recognized objects that appeared to be the same and yet appeared different in our similarity analysis. For example, object names of '*Aaron Neville and Linda Ronstad - I Don't Know Much.mp3*', '*Aaron Neville and linda ronstandt- I Don't Know Much.MP3*', '*Aaron Neville - Don't Know Much.mp3*', '*Aaron Neville ft. Linda Ronstadt - I Don't Know Much.mp3*' and '*Aaron Neville- I Don't Know Much (But I Know I Love You).MP3*' appeared to refer to the same object. Non-specific object names such as '*01 Track 1.wma*' appeared in 2,618 peers but was unlikely to be a replica of the same object. Perhaps we require sophisticated spelling correction operations [13] to identify similar names. However, this approach is made harder because terms used in song names and user queries need not follow from standard dictionaries.

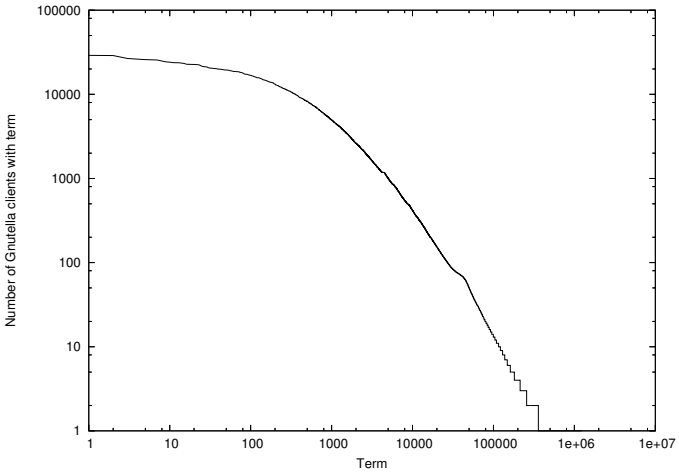


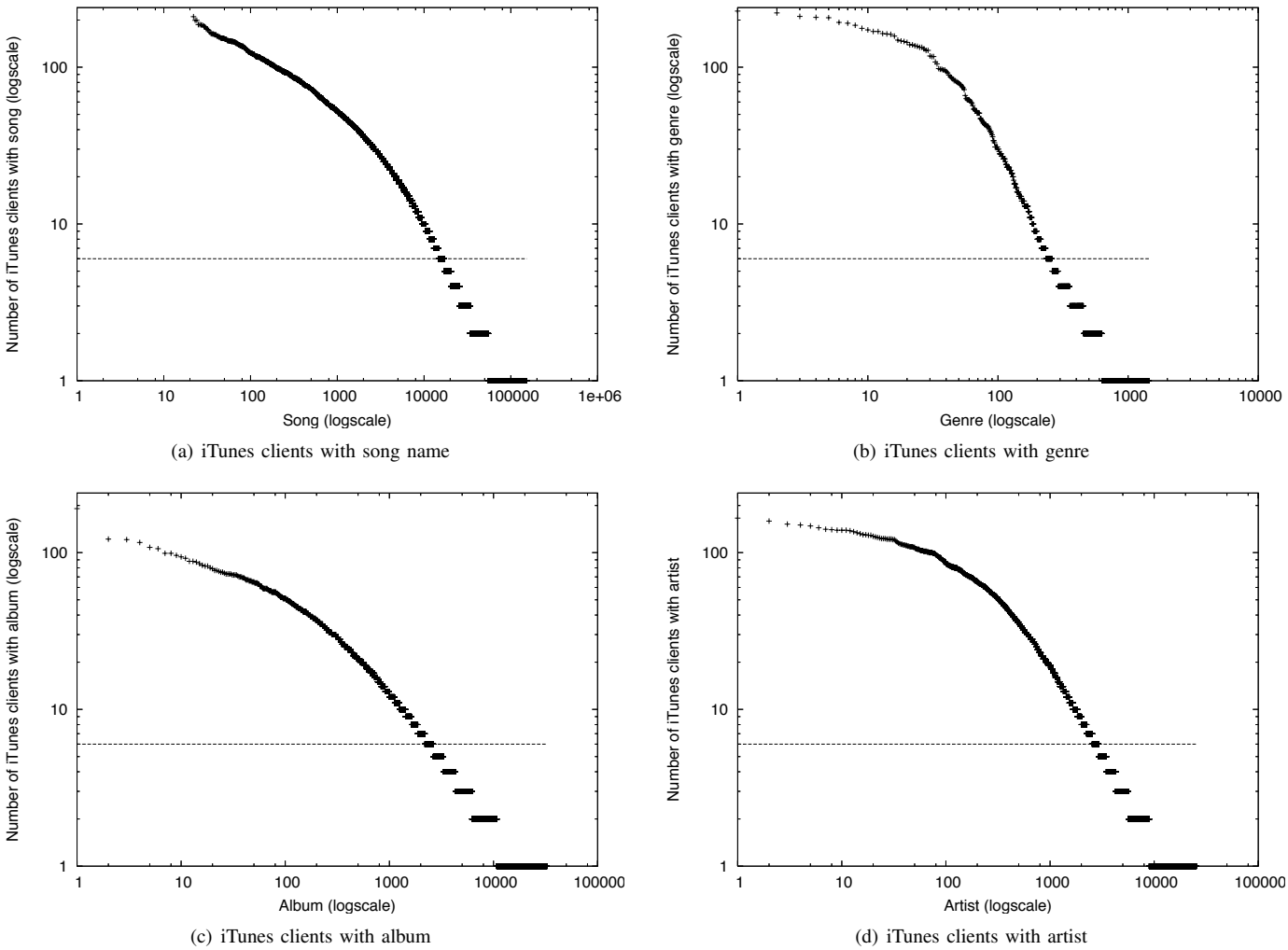
Fig. 3. Number of Gnutella clients with term(April, 2007)

Next, we split the terms in the Gnutella object names (using the Gnutella protocol tokenization mechanism) and plotted the cumulative distribution of the number of peers that contain each term of the name in Figure 3. From the graph, we noted that there were about 1.22 million unique terms. About 0.87 million terms (71.3%) were available in just one peer while 1.20 million terms (98.3%) were available in 37 or fewer nodes (less than 0.1% replication [14]). Both the entire filename and the individual terms (annotations) followed a Zipf distribution. This suggested that many objects and many terms were rare and thus making unstructured searches for them difficult.

B. Object annotations in iTunes

iTunes allowed the users to specify annotations such as song name, genre, artist, album etc. Next, we analyzed these object annotations. Our traces included 533,768 objects of which 171,068 objects were unique. These data traces were also used in an earlier paper [15].

We analyzed the popularity distribution of the song names in Figure 4(a). Of the 152,850 unique songs analyzed, 97,943 songs (64%) were available in just a single client. Similar to Gnutella networks, non-specific song names such as *TRACK* $< number >$ and *INTRO* appeared more than once in several



(a) iTunes clients with song name

(b) iTunes clients with genre

(c) iTunes clients with album

(d) iTunes clients with artist

Fig. 4. Analyzing annotations in iTunes

clients. Next we analyzed the distribution of the song genre (Figure 4(b)). iTunes was shipped with 24 genres. Users were free to change the genre name. Figure 4(b) showed about 1,452 genres. 99,987 songs (18.7%) did not have any genre. 817 genres (about 56%) occurred in a single peer. Next we analyzed the distribution of album names (Figure 4(c)). Album names were decided by the artist. Album names for songs purchased from iTunes store or ripped in the client will have these values from Gracenote. We observed 32,353 unique albums. 43,540 (about 8.1%) songs did not have an album name. About 21,250 (65.7%) were not replicated in any other peers. Finally, we analyzed the songs based on their artists (Figure 4(d)). Note that a song can be performed by many artists. Our traces showed songs from 25,309 unique artists. 16,500 artists (65%) of the artists appeared in a single peer. All of these annotations appeared to follow a Zipf distribution.

IV. Understanding the Properties of Search Terms and File Annotations

In the previous sections we showed that the the Zipf properties of the file annotations spans both different systems (Gnutella and iTunes) as well as different features of the annotations (e.g., genre, artist, album). In this section, we examined the characteristics of a real-world query workload as they relate to the file annotations.

An important assumption about unstructured P2P systems is that they can locate popular content easily. However, there is little overall correlation between the relative popularity of the query terms and the terms used in the file annotations [6]. This suggests that query performance in unstructured P2P systems is limited by the ability of the system to match popular query terms to file term annotations.

We were interested in determining how the relative popularity of query terms and file annotation terms varied over time. However, evaluating the change over time is complex. First, identifying which terms are popular requires a consistent def-

inition of popularity. Relying only on raw occurrence counts overlooked "hot" terms that become popular but whose raw occurrence counts may have been low over the long term. Due to this effect, we considered both raw popular and temporarily terms terms. This had a cascading effect on our examination of the relationship between popular query terms and popular file terms. We had to therefore examine the relationship between popular query and file terms with respect to the time-varying nature of popular query terms.

In the following sections we examine how the popularity of query terms changes over time. We then examine the similarity between popular query terms and file annotation terms over time.

A. Identifying Popular Query Terms

In this section, we were interested in determining the stability of popular query terms over time. We consider two types of popular terms. First, persistently popular terms are those terms that remained popular over the long term duration of our analysis. Second, transiently popular terms were those terms that exhibited significant deviation from their historical popularity for a given time interval. We were interested in identifying both classes of popular terms.

First, we examined the set of transiently popular terms. To identify these transiently popular terms, we first analyzed the occurrence of each terms using a training set of queries (10%) of the total queries. We then evaluated the popularity of each term at various evaluation intervals and compared the number of occurrences for each term during that interval to their historical number of occurrences. Terms that deviated significantly from their historical average were considered to be transiently popular for the evaluation interval.

Figure 5 plots the number of transiently popular terms over time for different evaluation intervals. Our experiments showed that although the mean number of transiently popular terms was low (< 10), there was significant variance in the number of of these transiently popular terms over the range of our evaluation intervals. We make two observations from these results. First, a large variance in transiently popular terms requires that the relationship between popular query terms and popular file terms take into account the time-varying nature of popular query terms. Second, the bulk of popular terms appear to be persistently popular as there are few transiently popular terms during each evaluation interval. We next examined the stability of the popular query terms over time.

B. Stability of the Set of Popular Query Terms

A set of popular query terms that varied little over time allowed for more opportunities for the P2P system to resolve the popular queries. We used the Jaccard index to analyze the similarity of the set of popular query terms over time. 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). Given two sets A and B, the Jaccard index of A and B is the ratio of the size of the intersection of A and B to the size of the union of A and B:

$$Jaccard(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

During a given time interval t, Q_t is the set of query terms observed during the interval and Q'_t is the set of popular query terms for the interval. We then define $Q''_t = Q'_t \cap Q'_{t-1}$, the set of persistently popular query terms at time interval t, as the set of popular query terms for time interval t that were also popular during the previous interval $(t-1)$. We computed $Jaccard(Q'_t, Q'_{t-1})$ at each evaluation interval.

Figure 6 plots the Jaccard similarity between the popular query terms for an interval and the terms that were popular in the prior interval. The analysis was performed on a one week query trace using a 60 minute evaluation interval. Although we only present results for 60 minute evaluation intervals, we witnessed consistent results across the different evaluation intervals. In Figure 6, we see that after a short stabilization time[1], the set of popular terms remains relatively constant as exhibited by the high Jaccard similarity value ($> 90\%$). This suggested a stable set of popular query terms that can potentially allow for more opportunities to resolve popular queries.

C. The Disconnect Between Query Terms and File Terms

In this section, we were interested in determining whether the shared file terms correspond well to the set of observed query terms. From our query trace and crawl data we identified the set of popular file terms (F) and the set of popular query terms for a given evaluation interval (Q_t). Using the Jaccard index ($Jaccard(F, Q_t)$), we observed only a 5% similarity between the query terms for the interval and the terms of all shared objects.

Figure 7 plots the Jaccard similarity over time between the set of query terms for the given interval and the set of popular file terms. We see that the similarity between query terms and file terms remained low ($< 20\%$) for all evaluation interval values. These results suggested that there was a disconnect between the file terms and query terms; the terms used to describe the objects were not the same as the terms that users specified in the queries to locate the shared objects. This disconnect is important because it suggests that despite having a stable population of query terms over time, there was little opportunity for queries with popular terms to be resolved against files in the system. In the following section we discuss some implications of our findings with respect to query performance.

V. Implications of these observations

The trend in both unstructured and hybrid P2P networks was toward lower TTL values. Hybrid P2P systems preferred lower TTL values in order to rapidly identify rare queries (which can be queried using a structured overlay). Unstructured P2P

[1]Because this analysis compared data from one interval to data from the previous interval, the first few intervals exhibited significant variance as the overall popularity counts for many terms had yet to be established.

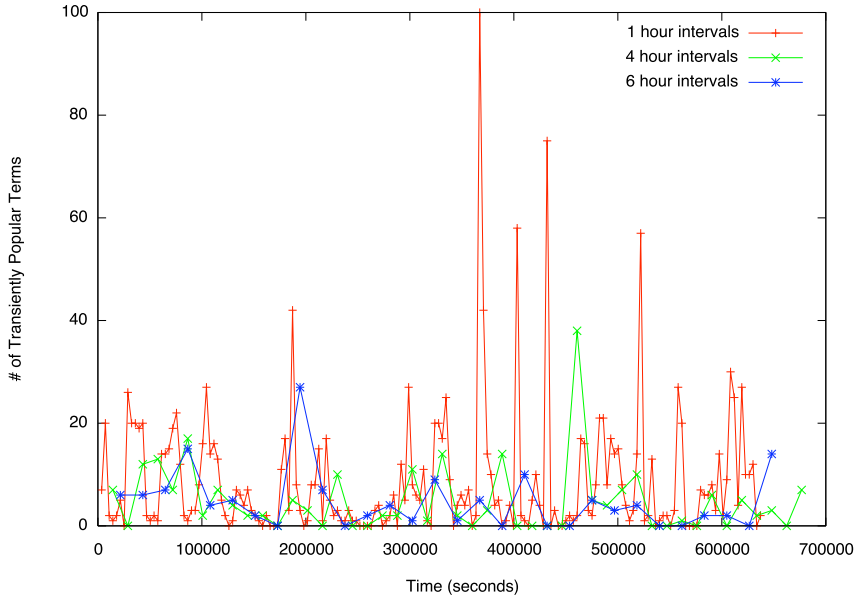


Fig. 5. Number of transiently popular query terms vs. time for different evaluation intervals. The overall mean was low, but there was significant variance across evaluation intervals.

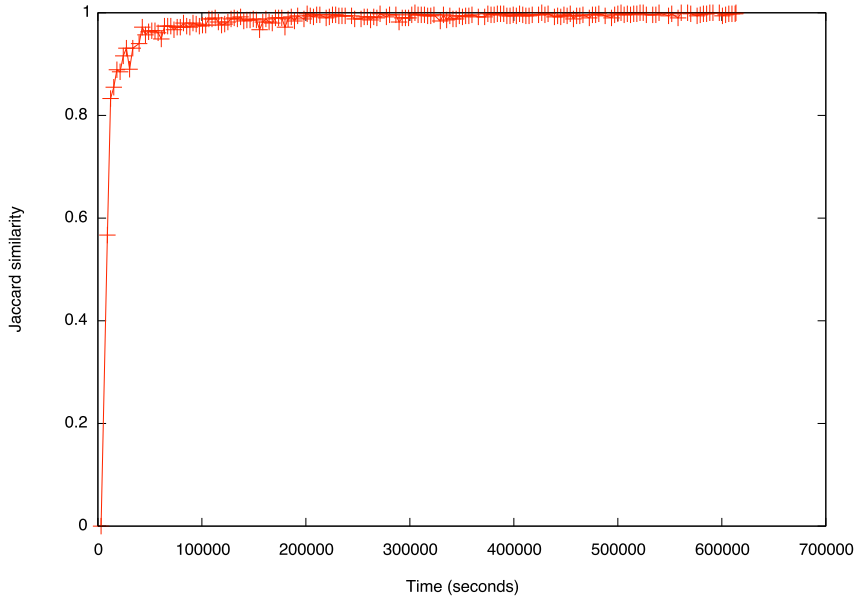


Fig. 6. Jaccard similarity over time for the set of popular terms (Q'_t) vs. the set of previously popular query terms (Q'_{t-1}). Analysis performed on a one week query trace with the evaluation interval set to 60 minutes.

systems preferred lower TTL values in order to restrict the reach of queries into the network. In 2006, the mean number of hops taken for Gnutella queries arriving at a monitoring peer was 2.47 hops [16]. In order to fully understand the true implications of our Zipf observation, we conducted a simple simulation. We varied the query TTL and compared the query success rates for a 40,000 node Gnutella network that distributed the objects either in a uniformly random fashion or as a Zipf distribution. For the uniformly distributed scenario, we varied the number of replicas available by 1, 4, 9, 19 and 39 for a corresponding replication ratios of 0.005%, 0.0125%, 0.025%, 0.05% and 0.1%, respectively. The average number of replicas of our dataset was 5. For each of the TTL values of 1, 2, 3, 4 and 5, on average the query reached 0.015%, 0.2525%, 3.0675%, 26.25% and 82.95% of the peers, respectively. We noted that the Zipf distribution behaved similar to a system with replication ratios as low as 0.005%. Hybrid P2P systems are likely to choose a small TTL before switching to structured lookup. In a system that used a TTL of 3, the query reached over a thousand nodes and yet achieved a success rate of

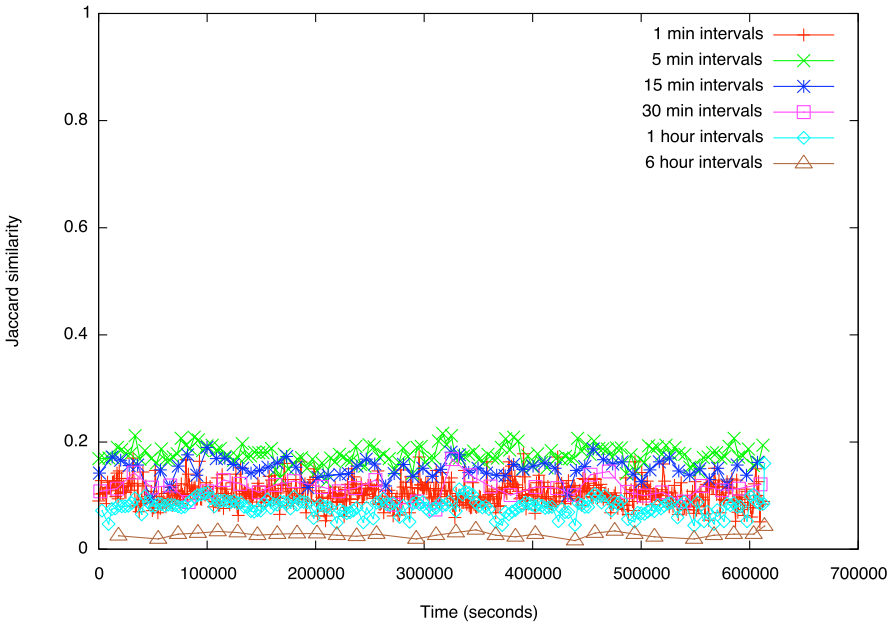


Fig. 7. Jaccard similarity over time for the set of query terms for the time interval (Q_t) vs. the set of popular file terms (F').

about 5%. For comparison, a random distribution model with a replication ratio of 0.1%, would have predicted a success rate of 62%. Our analysis showed that very few objects actually have a high enough replication ratio under the power law distribution of object replicas to make the flooding component of hybrid P2P searches feasible.

VI. Related work

Gia [17] employed one-hop replication, topology adaptation and a search mechanism based on random walks to improve the search performance of Gnutella. Gia was evaluated using a uniform object distribution on up to 0.5% of the peers. We show that the Zipf distribution exhibited in real-world P2P systems located fewer than 1% of the objects with replication ratios as high as 0.5%.

Zaharia et al. [13] showed that 20% of the file descriptions were misspelt. Our analysis showed that the annotations themselves exhibited a Zipf like behavior. Fessant et al. [18] analyzed the eDonkey network to show that the objects exhibited a Zipf like distribution. Similarly Zhao et al. [19] illustrated that objects exhibited a Zipf distribution. We show that the annotations were Zipf like for a wide variety of object annotations and this distribution significantly affected the query performance.

Hybrid P2P [5], [20], [21] systems had leveraged the combined benefits of unstructured and structured P2P systems to efficiently locate both popular and rare contents. Queries were first flooded using the unstructured network to locate popular contents. If the search failed, then the query was re-issued using the structured mechanism. However, we showed that few contents had a sufficiently high replication ratio in order to make the unstructured component of hybrid P2P systems feasible. In [5], Loo et al. considered queries to be rare if they contained fewer than 20 results. Our results show that fewer than 4% of the objects in the system are replicated on 20 or more peers.

VII. Discussion

There is a need to develop unstructured overlays that are aware of underlying temporal distributions of objects queries. We showed that only 2% of objects were replicated in more than 0.1% of the nodes can be popular. A hybrid P2P system that used this observed object distribution would perform worse than a DHT-based search because few objects are replicated enough to make the unstructured search component efficient. We showed that the set of popular query terms remains stable over time and exhibit a similarity of over 90%. We also showed that despite the Zipf popularity distributions of both query terms and file annotation terms, there is little similarity over time ($< 20\%$) between popular file annotation terms and popular file terms. Prior P2P search performance analysis did not take into account this mismatch between the query terms and object annotations and thus overestimated query performance. We believe that our observation requires a rethinking of unstructured P2P mechanisms for practical networks. Our ongoing research is focused on building P2P systems that can react to the observed temporal changes in query term popularity [9]. Our approach created synopses of content at each peer. The synopses was adapted dynamically to take into account transiently popular terms and thus improved overall search success rates.

Acknowledgments

Supported in part by the U.S. National Science Foundation through grant CNS-0447671.

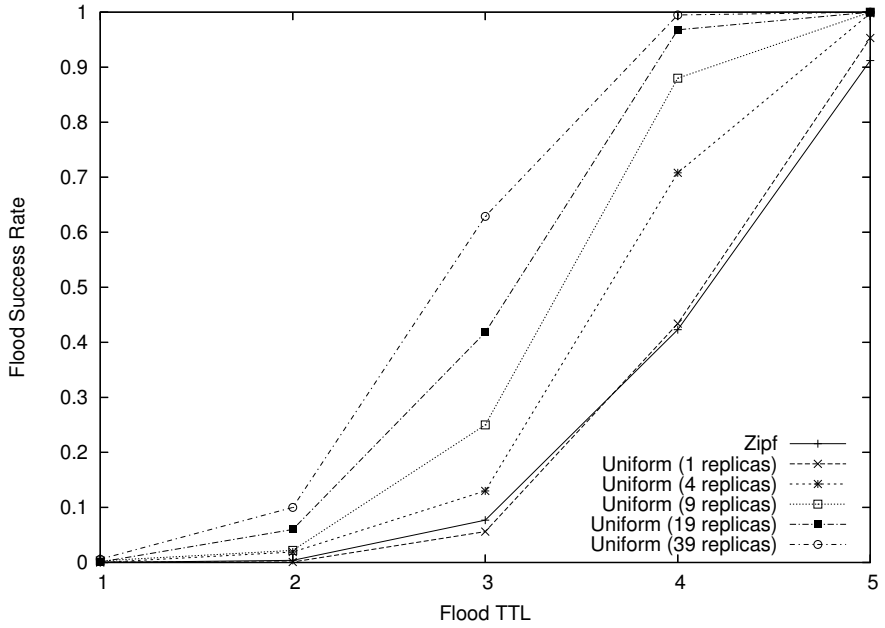


Fig. 8. Query success rates for a 40,000 node Gnutella network using uniform distribution as well as Zipf distribution. Replication amounts were varied.

REFERENCES

[1] A. Rowstron and P. Drushel, "Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems," in *18th IFIP/ACM Conference on Distributed Systems Platforms (Middleware 2001)*, Heidelberg, Germany, Nov. 2001.

[2] B. Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, A. D. Joseph, and J. D. Kubiatowicz, "Tapestry: A resilient global-scale overlay for service deployment," *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 1, pp. 41–53, Jan. 2004.

[3] "The gnutella protocol specification v0.6," http://rfc-gnutella.sourceforge.net/src/rfc-0_6-draft.html.

[4] L. Massoulie, E. L. Merrer, A.-M. Kermarrec, and A. Ganesh, "Peer counting and sampling in overlay networks: random walk methods," in *PODC '06: Proceedings of the twenty-fifth annual ACM symposium on Principles of distributed computing*, Jul. 2006, pp. 123–132.

[5] B. T. Loo, R. Huebsch, I. Stoica, and J. M. Hellerstein, "The case for a hybrid p2p search infrastructure," in *Workshop on Peer-to-Peer Systems (IPTPS '04)*, San Diego, CA, Feb. 2004, pp. 141–150.

[6] 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.

[7] S. Zhao, D. Stutzbach, and R. Rejaie, "Characterizing files in the modern gnutella network: A measurement study," in *Proceedings of the SPIE/ACM Multimedia Computing and Networking (MMCN '06)*, San Jose, CA, 2006.

[8] H. Pucha, D. G. Andersen, and M. Kaminsky, "Exploiting similarity for multi-source downloads using file handprints," in *Proc. 4th USENIX NSDI*, Cambridge, MA, Apr. 2007.

[9] W. Acosta and S. Chandra, "Exploiting the properties of query workload and file name distributions to improve p2p synopsis-based searches," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM '08) - Mini Symposium*, Phoenix, AZ, April 2008.

[10] D. Stutzbach and R. Rejaie, "Capturing accurate snapshots of the gnutella network," in *IEEE Global Internet Symposium*, Miami, FL, Mar. 2005, pp. 127–132.

[11] "The phex gnutella client," http://phex.kouk.de.

[12] C. Davies, "Applerecords," www.cdavies.org/applerecords.html.

[13] M. A. Zaharia, A. Chandel, S. Saroiu, and S. Keshav, "Finding content in file-sharing networks when you can't even spell," in *6th Workshop on Peer-to-Peer Systems (IPTPS'07)*, Bellevue, WA, Feb. 2007.

[14] W. Acosta and S. Chandra, "Improving search using a fault-tolerant overlay in unstructured p2p systems," in *36th International Conference on Parallel Processing (ICPP '07)*, Xian, China, Sep. 2007.

[15] S. Chandra and X. Yu, "Share with thy neighbors," in *Multimedia Computing and Networking (MMCN 2007)*, San Jose, CA, Jan. 2007.

[16] W. Acosta and S. Chandra, "Trace driven analysis of the long term evolution of gnutella peer-to-peer traffic," in *the eighth Passive and Active Measurement conference (PAM 2007)*, Louvain-la-neuve, Belgium, Apr. 2007.

[17] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, and S. Shenker, "Making gnutella-like p2p systems scalable," in *Applications, technologies, architectures, and protocols for computer communications (SIGCOMM '03)*, 2003, pp. 407–418.

[18] F. L. Fessant, S. Handurukande, A.-M. Kermarrec, and L. Massouli, "Clustering in peer-to-peer file sharing workloads," in *Proceedings of the 3rd International Workshop on Peer-to-Peer Systems (IPTPS'04)*, San Diego, CA, Feb. 2004.

[19] S. Zhao, D. Stutzbach, and R. Rejaie, "Characterizing files in the modern gnutella network: A measurement study," in *Proceedings of SPIE/ACM Multimedia Computing and Networking*, vol. 6071, San Jose, CA, Jan. 2006.

[20] M. Zaharia and S. Keshav, "Gossip-based search selection in hybrid peer-to-peer networks," in *5th Workshop on Peer-to-Peer Systems (IPTPS '06)*, Santa Barbara, CA, Feb. 2006.

[21] 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)*, Xian, China, Sep. 2007.