This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE P2P 2012 proceedings

An Efficient and Trustworthy P2P and Social Network Integrated File Sharing System

Guoxin Liu, Haiying Shen
The Department of Electrical and Computer Engineering
Clemson University, Clemson, SC 29631
Email: {guoxinl, shenh}@clemson.edu

Lee Ward
Sandia national laboratories
Albuquerque, New Mexico 87185
Email:lee@sandia.gov

***Abstract*—Efficient and trustworthy file querying is important to the overall performance of peer-to-peer (P2P) file sharing systems. Emerging methods are beginning to address this challenge by exploiting online social networks (OSNs). However, current OSN-based methods simply cluster common-interest nodes for high efficiency or limit the interaction between social friends for high trustworthiness, which provides limited enhancement or contradicts the open and free service goal of P2P systems. Little research has been undertaken to fully and cooperatively leverage OSNs with integrated consideration of proximity and interest. In this work, we analyze a BitTorrent file sharing trace, which proves the necessity of proximity- and interest-aware clustering. Based on the trace study and OSN properties, we propose a SOcial Network integrated P2P file sharing system with enhanced Efficiency and Trustworthiness (SoNet) to fully and cooperatively leverage the common-interest, proximity-close and trust properties of OSN friends. SoNet uses a hierarchical distributed hash table (DHT) to cluster common-interest nodes, then further cluster proximity-close nodes into subcluster, and connects the nodes in a subcluster with social links. Thus, when queries travel along trustable social links, they also gain higher probability of being successfully resolved by proximity-close nodes, simultaneously enhancing efficiency and trustworthiness. The results of trace-driven experiments on the real-world PlanetLab testbed demonstrate the higher efficiency and trustworthiness of SoNet compared with other systems.**

I. Introduction

Advancements in technology over the past decade have stimulated the development of large-scale peer-to-peer (P2P) file sharing systems where globally scattered nodes interconnect to realize collaborative file services. Providing both highly efficient and trustworthy service is perhaps one of the more formidable challenges facing large-scale P2P system research. In such a system, millions of nodes are scattered worldwide across disparate administrative domains. The large-scale, global node distribution and dynamism of the system dramatically increase the difficulty of providing efficient data querying that would allow nodes to quickly receive queried data at low cost. Also, distributed and autonomous users without preexisting trust relationships in the system make it critical to ensure a trustworthy environment that prevents both selfish and malicious behaviors. Indeed, 45% of files downloaded through the Kazaa file sharing application contained malicious code [1], and 85% of Gnutella users were sharing no files [2]. A growing need persists for a highly efficient and trustworthy P2P file sharing system that can i) efficiently and trustworthily forward a query to file servers, and ii) effectively provide incentives to encourage nodes to be cooperative in providing files. However, in spite of the significant efforts to tackle the challenge of efficiency and trustworthiness, current methods are insufficiently effective. By "trustworthiness", we mean a peer's willingness to cooperate in forwarding and responding to a query.

To further improve efficiency, some works consider proximity [3–11], and some works cluster nodes based on node interests or file semantics [12–18]. However, most of these methods fail to simultaneously consider proximity (proximity and locality are interchangeable terms in this paper) and interest. Recently, emerging methods are beginning to address the challenge of high efficiency and trustworthiness by exploiting online social networks (OSNs). By leveraging the social property of "friendship fosters cooperation" [19] and common-interest, some OSN-based systems [20, 21] cluster common-interest OSN friends for high efficiency and trust, but they fail to leverage OSNs for proximity-aware search or efficient intra-cluster search. Though some OSN-based systems [22–25] use social links for trustworthy routing, they cannot guarantee data location. By only considering routing or data discovery between friends, these approaches cannot significantly enhance the efficiency and trustworthiness, and also contradict the open and free data sharing goal of large-scale P2P file sharing systems. Little research has been undertaken to fully and cooperatively leverage OSNs to significantly enhance the efficiency and trustworthiness of P2P file sharing systems. By "cooperatively", we mean that the OSN-based methods should coordinate with previous P2P methods to ensure the availably of search results, and not confine sharing among friends.

To address the problem, we propose a SOcial Network integrated P2P file sharing system for enhanced Efficiency and Trustworthiness (SoNet). SoNet fully and cooperatively leverages OSNs in designing advanced mechanisms based on OSN properties and our observations on the necessity of interest- and proximity-aware node clustering. By "integrated," we mean that an OSN is merged into a P2P system by using social links directly as overlay links, and by exploiting social properties in the technical design of the P2P system, rather than simply combining two separate systems such as the Maze file sharing system [26]. *SoNet is the first to build a hierarchical DHT to fully exploit the common-interest, proximity-close and trust properties of friends in OSNs for simultaneous interest/proximity-aware and trustworthy file querying.* The detailed contribution of this work can be summarized as below.

BitTorrent trace study. We analyze a BitTorrent trace to verify

978-1-4673-2862-3/12/$31.00 ©2012 IEEE

the importance of proximity- and interest-aware clustering and its integration with OSN friend clustering, and file replication.

A social-integrated DHT. SoNet novelly incorporates a hierarchical DHT to cluster common-interest nodes, then further clusters proximity-close nodes into subcluster, and connects the nodes in a subcluster with social links.

Efficient and trustworthy data querying. When queries travel along trustable social links, they also gain higher probability of being successfully resolved by proximity-close nodes. Unsolved queries can be resolved in an interest cluster by proximity-close nodes for system-wide free file querying.

Social based query path selection. Since common sub-interest (subclass of interest classification, e.g., country music within music) nodes within a larger interest tend to connect together, in the social link querying, a requester chooses K paths with the highest past success rates and lowest latencies based on its query's sub-interest.

Follower and cluster based file replication. A node replicates its newly created files to its followers (interest-followers) that have visited majority of its files (files in the created file's interest). Also, frequently visited file between subclusters and clusters are replicated for efficient file retrieval.

The rest of this paper is structured as follows. Section II presents a concise review of related work. Section III presents OSN properties utilized by SoNet and our study on a BitTorrent trace. Section IV details the design of SoNet. Section V shows the performance of SoNet compared with other systems in trace-driven experiments on PlanetLab [27]. Section VI concludes this paper with remarks on our future work.

II. Related work

In order to enhance the efficiency of P2P file sharing systems, some works cluster nodes based on node interest or file semantics [12–18]. Iamnitchi *et al.* [12] found the smallworld pattern in the interest-sharing community graphs, and suggested clustering common-interest nodes to improve file searching efficiency. Li *et al.* [13] clustered peers having semantically similar data into communities, and found the smallworld property from the clustering, which can be leveraged to enhance the efficiency of intra- and inter-cluster querying. Chen *et al.* [14] built a search protocol routing through users having common interests to improve searching performance. Lin *et al.* [15] proposed a social based P2P assisted video sharing system through friends and acquaintances, which can alleviate the traffic of servers and share videos efficiently. Chen *et al.* [16] constructed a P2P overlay by clustering common-interest users to support efficient short video sharing. Li *et al.* [17] grouped users by interests for efficient file querying and used the relevant judgment of a file to a query to facilitate subsequent same queries. Shen *et al.* [18] proposed a multi-attribute range query method with locality-awareness for efficient file searching.

Some works improve the searching efficiency with proximity-awareness. Genaud *et al.* [3] proposed a P2P-based middleware, called P2P-MPI, for proximity-aware resource discovery. Liu *et al.* [4] took PPLive as an example and examined traffic locality in Internet P2P streaming systems. Shen and Hwang [5] proposed a locality-aware architecture with resource clustering and discovery algorithms for efficient and robust resource discovery in wide-area distributed grid systems. Yang *et al.* [6] combined the structured and unstructured overlay with proximity-awareness for P2P networks, and the central-core structured overlay with supernodes insures the availability of searching results. A number of other works with proximity-awareness also take into account the physical structure of the underlying network [7–11]. However, most of the proximity-aware and interest-clustering works fail to simultaneously consider both proximity and interest, and trustworthiness of file searching.

Social links among friends in OSNs are trustable and altruistic [19], which can further facilitate the efficiency and trustworthiness of data searching. Some OSN-based systems cluster common-interest OSN friends for high efficiency and trustworthiness [20, 21]. However, these works fail to further leverage OSNs for efficient intra-cluster search and proximity-aware search. A number of other OSN-based systems use social links for trustworthy routing [22–25]. However, they either only use social links to complement the DHT routing [22, 23], which provides limited efficiency enhancement, or directly regard an OSN as an overlay [24, 25], which cannot guarantee data location.

SoNet shares similarity with the works [5, 6, 28–30] in utilizing supernodes with high capacity to enhance file searching efficiency. Different from current works, SoNet is the first P2P system that fully and cooperatively leverages the properties of OSNs to integrate with the proximity- and interest-clustering of nodes in a DHT for high efficiency and trustworthiness. To leverage trustworthiness inside OSNs, any works exploiting trust relationships for access control in OSNs [31] are orthogonal to our study.

III. BitTorrent Trace Data Study

A. Observations from OSNs

In OSNs, nodes with close social relationships tend to have common interests [32] and tend to be located in the same place [33]. These observations are confirmed by a study on the video sharing in the Facebook OSN [34] which revealed that i) around 90% of a video's viewers are within two social hops of the video owner, ii) most viewers of a video are in the same city of the video owner, and iii) users tend to watch videos within their interests. In a nutshell, nodes in OSNs tend to visit files within their interests and from socially close nodes (proximity-close and common-interest). Therefore, we arrive at a conclusion (C):

C1: *The interest/proximity-awareness feature proves the necessity of OSN friend-clustering, in which efficient data queries transmit though social links as logical links.*

Trust queries as well as recommendations can travel though Social links, which can be applied in social-based file sharing system and question-answer system. However, a node's queried data within its interests may not be held by its socially close nodes. A logical overlay that cooperatively merges the social links is needed for open, free and deterministic data querying. We thus seek to determine the feasibility of interest/proximity node clustering in a P2P file sharing system:

do nodes in a location share data in a few interests? If yes, we can use interest/proximity-aware clustering that maps OSN friend-clustering to complement social link querying. Through our study on the BitTorrent trace below, we arrive at a positive answer for the above question.

B. BitTorrent Trace Study

The BitTorrent User Activity Trace [35] traced the downloading status of 3,570,587 peers in 242 countries requesting for 36,075 files in 366 file categories. We regarded file categories such as Comedy, Sports, and Animation as different file interests. We regarded a node's country as its location and grouped nodes by their locations. The trace does not provide the information of the servers for a requested file of a client. Since there are five main downloading connections for a peer's file request according to the uplink utilization strategy in BitTorrent [36], we randomly chose 5 servers that were uploading a client's requested file during the same time period when the client is downloading the file.

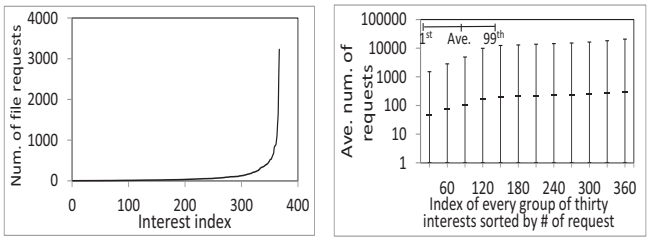
C. Necessity of Proximity-aware Clustering

We measured the distance of each pair of two different countries. The average, maximum, and minimum distances between all pairs of countries are 8518km, 19903km and 39km, respectively. We then measured the distance between each pair of file provider and requester for a file request and used the average of the five pairs of the request as its requester-provider distance.

Figure 1(a) shows the cumulative distribution function (CDF) for the percent of file requests versus the requester-provider distance. Nearly 50% of the file requesters retrieve files from providers that are more than 9000km away. Also, only 10% of the files can be retrieved from providers that are less than 3000km away. We calculated that the average requester-provider distance is around 7500km, which equals to the average distance of all pairs of nodes. The long distance greatly increases the cost of file retrieval. In Figure 1(b), the x axis stands for the number of requester-provider pairs of a file, and the y axis represents the average distance of all pairs of that file. This figure indicates that most files are retrieved from nodes with long distance within [2500,7500]km.

We use S to denote the set of all countries and R_{ij} to denote the number of requests from country i to country j. We define country i's *country request coefficient* as $C_r(i) = R_{ii}/\sum R_{ij}$ $(j \in S)$, which means the percentage of requests within country i. Figure 1(c) shows the C_r distribution over all countries. We see that 80% of countries have $\leq$0.02 country coefficient, 90% of countries have $\leq$0.04 country coefficient, and 99.5% of countries have $\leq$0.5 country coefficient. The result shows that nodes in most countries access files in other countries rather than in their own countries. This implies that the percentage of requests responded by local providers (in the same location) is very low without a locality-aware strategy, and peers choose non-local providers (not in the same location) with high probability. This verifies the importance of proximity-awareness in file searching.

We use N to denote the number of files requested by the peers in a country. Multiple requests for the same file are counted as one. We use N_s to denote the number of files among the N files that are requested by at least one peer in another country and define the *sharing correlation* of a country as $C_{so} = N_s/N$. Figure 1(d) shows the sharing correlations for each country, most C_{so} are 100% or very close to 100%. This means nearly all the files in one country are visited by other countries.



(a) Distribution of files over interests (b) Distribution of peers over countries and interests

Fig. 4: Distribution of interests.

C2: *The long requester-provider distances and remote file retrievals in current file sharing system make the locality-aware file sharing desirable for enhanced file sharing efficiency.*

D. Necessity of Interest-based Clustering

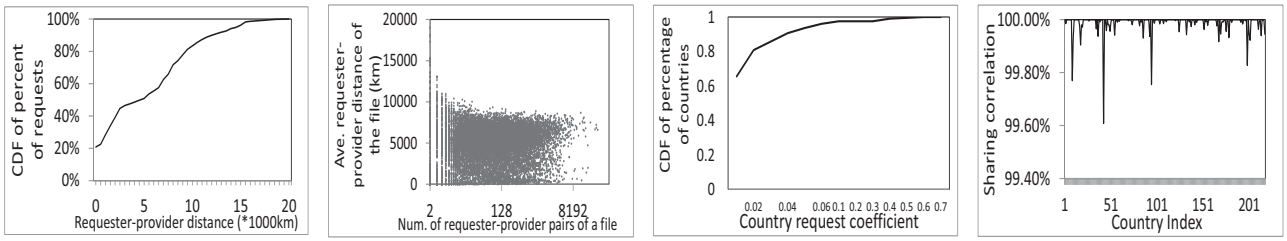
By "an interest requested by a peer," we mean "an interest whose files are requested by a peer." We use c to denote a country, and use $\mathcal{R}$ and $\mathcal{R}_c$ to denote the group of all interests requested by the peers in all the countries and in country c, respectively. For each country c, we calculated the number of requests for files in each interest denoted by $F_{i,c}$ $(i \in \mathcal{R}_c)$. We then calculated the average value of the numbers: $\bar{F}_c = \sum_{i \in \mathcal{R}_c} F_{i,c}/|\mathcal{R}_c|$ and regarded it as an interest threshold of the country. We then regarded the interest whose number of requests is above the threshold ($F_{i,c} \geq \bar{F}_c$) as the country's *main interest*, denoted by $\mathcal{I}_c$. For each country, we calculated the percentage of requests for the country's main interests in the country's total interests: $P_F = \sum_{i \in \mathcal{I}_c} F_{i,c} / \sum_{j \in \mathcal{R}_c} F_{j,c}$. We also calculated the percentage of the country's main interests in the number of total interests of all the countries: $P_N = |\mathcal{I}_c|/|\mathcal{R}|$.

Figures 2(a) and 2(b) plot the P_N and $\mathcal{I}_c$ versus the P_F for each country, respectively. The figure shows that in each country, more than 50% of file requests are for less than 15% of the total interests. Most countries' main interests constitute 10% of the total interests, and the requests in their main interests constitute 75%-85%. In some countries, even 100% of the file requests are focused on less than 5% of the total interests. The result indicates that the request distribution over interests approximately obeys a power-law distribution.

C3: *Nodes in a cluster tend to visit files in a few interests, which necessitates interest-based subcluster clustering.*

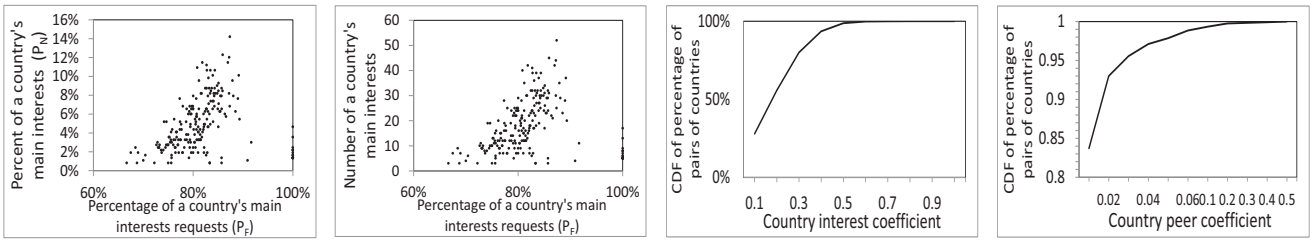
E. Cluster-based File Replication

With interest- and proximity-aware clustering, we define the *country interest coefficient* as $C_{\mathcal{I}} = |\mathcal{I}_i \cap \mathcal{I}_j|/|\mathcal{I}_i \cup \mathcal{I}_j|$. Figure 3(a) shows the CDF of the percentage of country pairs versus the country interest coefficient. We see that 28%



(a) Distribution of requester-provider pairs (b) Distance of file retrieval (c) Distribution of country request coefficient (d) File sharing among countries

Fig. 1: Necessity of locality-aware node clustering.



(a) Distribution of requests on main interests (b) The number of requests on main interests (a) Distribution of country pairs vs. country interest coefficient (b) Distribution of country peer coefficient

Fig. 2: Necessity of interest-based subcluster clustering.

Fig. 3: Necessity of cluster-based file replication.

of country pairs have a $\leq$0.1 coefficient, 60% of country pairs have a $\leq$0.2 coefficient, 80% of country pairs have a $\leq$0.3 coefficient, and approximately all countries have a $\leq$0.5 coefficient. The result shows that some nodes in different locations share the same interests.

C4: *The interest similarity between countries suggests that in order to enhance search efficiency, file replication needs to be executed between locations for popular files.*

Figure 4(a) plots the number of file requests in each interest in the entire trace data. The number of files' distribution over interests obeys the power-law distribution. Thus, some files have high popularity while others have low popularity during a certain time period.

For each interest (file category), we calculated the number of file requests from a country in the entire trace data. We sorted the interests in the ascending order by the average number of requests per country for each interest. Figure 4(b) shows the 1st percentile, the 99th percentile, and the average of the numbers for each group of 30 interests. We see that for each group, the 99th percentile is much larger than the average, and the average is much larger than the 1st percentile. Thus, a given file category has high popularity in some locations and low popularity in others. Finally, from Figures 4(a) and 4(b), we derive:

C5: *File replication is needed between locations for popular files in each interest, and system-wide file searching is needed for locating unpopular files.*

IV. SOCIAL NETWORK INTEGRATED P2P FILE SHARING SYSTEM

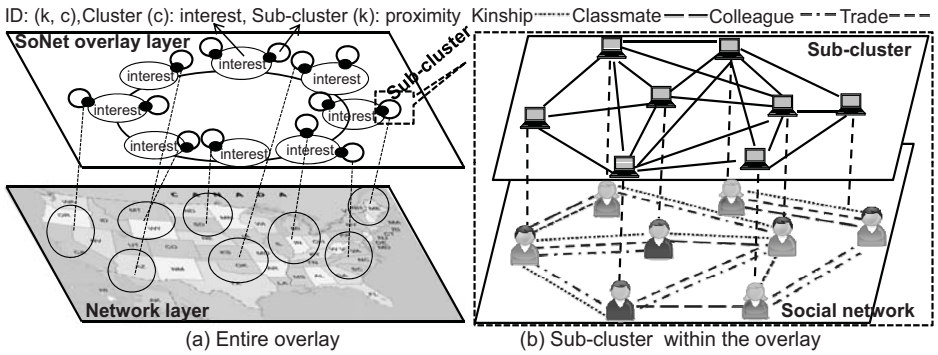
A. An overview of SoNet

Based on C1 and the social property of "friendship fosters cooperation" [19], SoNet directly uses social links as logical links for efficient and trustworthy data querying among socially close nodes. For open, free and deterministic system-wide data querying, SoNet uses interest/proximity-aware clustering that matches the OSN friend-clustering. For trustworthy file querying between non-friends, SoNet can employ reputation systems [37, 38] to provide cooperative incentives. We do not explain the details of the reputation systems as the techniques are orthogonal to our study in this paper.

According to C2, we cluster physically close nodes into a cluster. According to C3, we further group nodes in a cluster sharing a single interest into a subcluster. Since the high scalability, efficiency and deterministic data location make DHTs favorable overlays, SoNet aims to build a DHT embedded with interest/proximity-aware clusters and OSN friend clusters. According to C4 and C5, we propose a follower and cluster based file replication algorithm. SoNet is the first to fully and cooperatively exploit the properties of OSNs and DHTs, which enhances efficiency and trustworthiness simultaneously with consideration of both proximity and interest. Below, we introduce each component of SoNet.

B. A Social-integrated DHT

DHT overlays [39–41] are well-known for their high scalability and efficiency. *However, few previous works can cluster nodes based on both proximity and interest in a single DHT while integrating an OSN.* SoNet is designed based on the Cycloid [42] DHT overlay and supernode structure [5, 6, 28–30]. Cycloid is a hierarchical structured overlay with $n = d{*}2^d$ nodes, where d is its dimension. In Cycloid, each node is represented by a pair of indices (k, c), where $k \in [1, d]$ and $c \in [1, 2^d]$. k differentiates nodes in the same cluster, and c differentiates clusters in the network. Each cluster has a primary node with the largest k in node ID and a query always passes the primary nodes in inter-cluster routing. Thus, Cycloid supports the hierarchical clustering of nodes based on their interest and locality together in a single DHT. As shown in Figure 5, SoNet leverages a hierarchical infrastructure to simultaneously consider interest/proximity-awareness and social based clustering. SoNet groups nodes with similar interest



(a) Entire overlay (b) Sub-cluster within the overlay

Fig. 5: The SoNet overlay infrastructure.

into the same cluster, and further groups proximity-close nodes into the same subcluster, and then connects nodes within a subcluster using their friendship.

Representation of interest and proximity. SoNet requires a user to enter its interests in his/her profile when registering for the system based on a globally uniform attribute list such as "movie" and "music". A node's interests are then described by a set of attributes, which are translated to a set of real numbers using consistent hash functions [43] (e.g., SHA-1), denoted by $< S_1, S_2, \cdots >$. We employed a method to represent a node's physical location by a real number denoted by $\mathcal{H}$ [5]. The closeness of $\mathcal{H}$ values of different nodes denotes the closeness of these nodes in the network.

SoNet structure and maintenance. Recall that each node in Cyloid is represented by a Cycloid ID denoted by (k, c). We set the range of $\mathcal{H}$ to $[1, d]$ and set the range of S to $[1, 2^d]$. In SoNet, a node i with m interests has m IDs, denoted by $(\mathcal{H}_i, S_1)$, $\cdots$, $(\mathcal{H}_i, S_m)$. As shown in Figure 5(a), by connecting nodes based on their Cycloid IDs, common-interest nodes with the same S are clustered into a cluster, in which physically-close nodes with the same $\mathcal{H}$ are further clustered into a subcluster. Logically closer subclusters have closer proximity. As shown in Figure 5(b), nodes in a subcluster connect with each other by their social friend links. Node i exists in m subclusters of different interest clusters. All nodes in a subcluster elect a stable supernode that has the most social links with cluster members as their head in the subcluster. Each node reports its files' information to its head. The head maintains a record of subcluster members and their files. Thus, a file is recorded by all heads with one of the multiple interests of this file. The subcluster heads that form the Cycloid structure take the responsibility of DHT lookup functionality.

When node i joins in the SoNet system, it first generates its interest ID $(S_1, \cdots, S_m)$ and its proximity ID $\mathcal{H}_i$. It then generates its IDs $(\mathcal{H}_i, S_1)$, $\cdots$, $(\mathcal{H}_i, S_m)$. By using the Cycloid DHT node join algorithm, node i joins in the clusters of its interests and the subcluster in the cluster that has its physically close nodes. Node i then connects to the head of its subcluster. From the record in the head, node i locates its social friends in the subcluster and connects to them. If there is no cluster having an interest of node i or no subcluster with $\mathcal{H}_i$, node i becomes the first node of the cluster or subcluster. When node i leaves the SoNet system, it notifies its subcluster head and its social friends. The head removes the record of node i and its files. Its social friends remove the links to node i. If leaving node i is a subcluster head, it notifies all members in the subcluster to elect a new head and transfers its record to the new head. Users' interests are dynamic. When a node loses one of its interests, it will leave the subcluster of this interest; if a node has a new interest, it will join the according subcluster. To detect the overlay link disconnection due to node abrupt departures, each node periodically probes its neighbors including its subcluster head. If a node's probing fails, it assumes that the probed node has abruptly departed the system and updates its corresponding link. If a head is detected to have been abruptly departed, a new head is elected and all subcluster members again report their files to the new head.

C. Efficient and Trustworthy Data Querying

SoNet enables nodes to cache and share their visited files, which facilitates the rapid dissemination of files among interested friends. If a requester queries a file within its own interests, the file should be in its cluster. In intra-cluster querying, to leverage OSNs for trustworthy and efficient search, the requester first executes intra-subcluster querying to find the file in its proximity-close nodes, and then executes inter-subcluster querying.

In the intra-subcluster querying, the query is forwarded along social links to find file from the requester's common-interest and proximity-close nodes in a trustworthy manner. The requester sends its query with a TTL (Time to Live) to K friends selected by the social based query path selection algorithm (Section IV-D). If the selected friends do not have the queried file, they forward the query to the nodes in the specified paths or randomly chosen nodes until TTL=0. Upon receiving a query, a node checks whether it has the requested file. If the requester cannot find the file after TTL steps of social routing, it resorts to its head, which checks its file index of its subcluster. If the queried file exists in the subcluster, the head notifies the file holder to send the file to the requester. Otherwise, the intra-subcluster searching fails. Then, the head of node i launches the inter-subcluster querying.

Recall that the distance between subclusters represents the physical distance between the nodes in the subclusters. Thus, in order to find the queried file that is most physically close to the requester, the query is forwarded sequentially between subcluster heads. Specifically, the head of node i forwards the query to its successor subcluster head. The query receiver head then checks its record to find the matching record of requested

file. If the queried file exists, then it notifies the file holder in its subcluster to send the file to the requester. Otherwise, it continues to forward the query to its successor head. This process continues until the queried file is found or the head of node i receives the query (i.e., intra-cluster searching fails).

From C3, we know that users still have infrequent visits on files beyond their own interests. This implies that there exist a certain number of inter-cluster queries as cluster represents interest. When node i queries data with interest S which is not in its interests, it conducted an inter-cluster searching by DHT $Lookup(\mathcal{H}_i, S)$ function, where $\mathcal{H}_i$ is normalized Hilbert value of node i, and S is the interest of the queried file. After the head in the cluster of $(\mathcal{H}_i, S)$ receives the query, it launches an intra-cluster search. The receiver head searches the queried file in its file index and then searches nearby heads until finding the matching files. This inter-cluster search guarantees the availability of files, which is a necessary complementary policy for searching based on social relationship and locality awareness.

D. Social based Query Path Selection

In SoNet, the social graph is in a subcluster, which is constructed by nodes having the same interest, so social based querying is within the same cluster of one interest. An interest can be classified into a number of sub-interests. For example, Computer Engineering can be classified to Computer Networks, Computer Systems and so on. In a social network, nodes in a sub-interest group within a larger interest group have a higher probability of connecting with each other (e.g., Lab members majoring in Computer Systems) [32]. From C3, we know that users intend to visit files of several interests they visit frequently [44, 45]. Leveraging these two social network properties, we propose a method to enhance intra-subcluster querying along social links in SoNet.

We classify each interest into sub-interests. When a query is forwarded along the social links, each forwarder piggybacks its IP address with the query. As a result, the file holder can know the entire forwarding path and sends it with the file back to the requester. For each sub-interest S_k, a requester records the successful query paths and their response latency from the query's initial time to the response arrival time for each query. The record is in the form of 4-tuple $< S_k : P_j, v, t >$, where P_j the querying path; v denotes the query success rate of this path for queries in interest S_k, which is calculated by the percent of the appearance times of this path in all successful query paths for queries in interest S_k; and t is average latency of all successful responses of this path for queries in interest S_k.

In order to increase success rate of file querying, for each sub-interest S_k, a requester first sorts all paths by their success rate v in a decreasing order, and sorts paths with the same success rate by response latency t in an increasing order. Later on, when the requester queries a file in the sub-interest S_k, it selects the first K paths that have the highest success rate (v) and shorter response latency (t). If there are fewer than K paths for the sub-interest S_k, the requester randomly chooses the next hops from its social friends. Thus, these paths have a high probability of quickly forwarding the query toward the sub-interest cluster nodes containing the queried file and being willing to provide this queried file. This policy helps nodes choose low-latency paths toward the file holders and receive the file quickly and trustworthily.

E. Follower and Cluster based File Replication

In an OSN, a node visits files driven by both social relationship and interests [34]. For example, a node always visits its friend's files. We define a node that visits a certain high percentage of all files of node i as its *full-follower*; one that visits a certain high percentage of files in one interest of node i as its *interest-follower*. Each node i keeps track of the file visit activities from each of other nodes j, represented by $< j, p_{total}, p_{s1}, p_{s2}, \cdots >$, where p_{total} denotes the percent of all files in node i that node j visits and p_{sk} denotes the percent of all files in interest k in node i that node j visits in a unit of time period. When p_{total} reaches a predefined threshold, node i regards node j as its full-follower. When p_{sk} reaches a predefined threshold, node i regards node j as its interest-follower in interest k. A node pushes its newly created file to its full-followers and its interest-followers of the file's interest. Thus, the full-followers and interest-followers of node i can directly retrieve their desired files locally without the need to request, which enhances the efficiency of file retrieval.

Recall that subclusters represent different locations of nodes in one interest cluster, and if a file query cannot be resolved in a subcluster, it is passed through the subcluster heads sequentially. When there are many file queries passing through the subclusters from subcluster i to another subcluster j, we can build a bridge between the head of subcluster i and the head of subcluster j to avoid subsequent query passing to save the cost. Specifically, each subcluster head i keeps track of its file visit rate to each of other subclusters j on a file F, represented by $< j, F, v >$, where v denotes the file visit rate. If a head i finds that the accumulated visit rates of its subcluster nodes on a file F in subcluster j is higher than a pre-defined threshold, it generates a replica of the file in itself for local file retrieval. Thus, the queries for this file from head i's subcluster can be resolved locally without the need of subsequential query passing.

Recall that a node may query for a file outside of its interests and the query has to be forwarded using the lookup function in a system-wide manner. It is possible that many nodes from a cluster query for files in another cluster. Similarly, in this case, we can use file replication to reduce the cost due to system-wide routing. Recall that in Cycloid, an inter-cluster query passes through the primary nodes of clusters. Thus, each primary node keeps track of the inter-cluster activities of its cluster in the form of $< S, F, v >$, where S represents a cluster where queries are sent to, v means the visit rate on the file F in cluster S during a unit time. When v is larger than a predefined threshold, the primary node replicates the file. Later on, it can directly respond with the replicated file without the need to forward the inter-cluster query.

File replicas help to improve the file querying efficiency. However, when the visit rate of the replicas is low, the replicas may bring about higher cost than the benefits. Thus, when the visit rate of a replica decreases below a pre-defined threshold, the replica is deleted.

V. Trace-driven Performance Evaluation

In order to evaluate the performance of SoNet in comparison with other file sharing systems, we built prototypes of the systems on the PlanetLab [27] real-world distributed testbed. We randomly selected 350 nodes all over the world, and clustered them into 20 locations using the previously introduced Hilbert number method. For each PlanetLab node, we randomly selected a country in the BitTorrent trace and assigned the country's interests (as shown in Figure 2(b)) to the PlanetLab node.

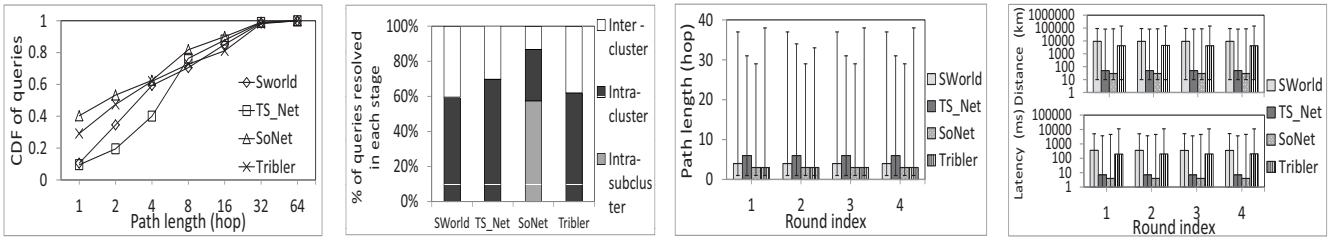
We set the dimension of Cycloid to 20. The system has 100,000 peers and used 366 interests from the BitTorrent trace. Each peer was assigned to a location randomly chosen from the 20 locations, was mapped to a randomly chosen PlanetLab node in the location, and was assigned 20% of the PlanetLab node's interests as its own interests. All peers mapped to the same PlanetLab node are 10km distant from each other. Each peer randomly selected 100 other peers as its friends that have at least one same interest, and the distribution of its friend over distance obeys power-law distribution [46]. The requests of a peer over interests follow the distribution as indicated in Figure 2(b), and the TTL of searching among social friends or common-interest nodes was set to 4 considering that a file can be discovered within 2 hops on average in a common-interest node cluster [12]. Each peer in SoNet maintains five social based query paths. In each experiment round, each peer generates a query sequentially at the rate of totally 10 queries per second in the system. We used the 36075 files in the BitTorrent trace and the files are randomly distributed among peers with the files' interests. 80% of all queries of a requester are located in peers within 4 social hops of the requester, and 70% of its queries are in the interests of the requester [12]. We also let each file have a copy owned by another peer in a different location in order to show the proximity-aware performance.

SoNet integrates interest/proximity-aware clustering and OSN friend clustering, while other systems leverage single clustering. Thus, we compared SoNet with three other systems with single clustering denoted by SWorld, TS_Net and Tribler that are variations of the systems in [12], [6] and [20], respectively. We modified the three systems to make them comparable to SoNet. In order to guarantee the success of file lookups, we complement the three systems with system-wide file lookups. That is, the file metadata is distributed using the Cycloid *Insert(ID,metadata)* function, and a file can always be found using the *Lookup(ID)* function. We use SWorld as a representative of interest-aware clustering systems. Its structure is the same as SoNet except that each peer in an interest cluster randomly selected 20 cluster peers to connect with and there are no proximity-aware subclusters. When a node queries for a file, it chooses K friends for K-multicasting with TTL=4 in its cluster. That is, each query receiver forwards the query to K randomly chosen neighbors until TTL=0. If the lookup fails, it uses *Lookup(ID)* to find the file. We use TS_Net as a representative of proximity-aware clustering systems. We use Cycloid as TS_Net's central structured overlay called T-network. We use 350 PlanetLab nodes to represent 350 different locations, and the peers in a location (mapped to a PlanetLab node) form a Cycloid cluster. The peers in a cluster form a four-ary proximity-aware tree [6] called S-network. We randomly selected 20 peers from each cluster as Cycloid peers. When a node queries for a file, it first searches the file in its tree in its cluster, and then uses *Lookup(ID)* to find the file. We use Tribler to represent the OSN-based file searching systems. Tribler directly connects peers using their social links and also builds the DHT overlay as SoNet. When a node queries for a file, it first randomly chooses K friends for K-multicasting with TTL=4, and then uses *Lookup(ID)* to find the file.

A. The Efficiency of File Searching

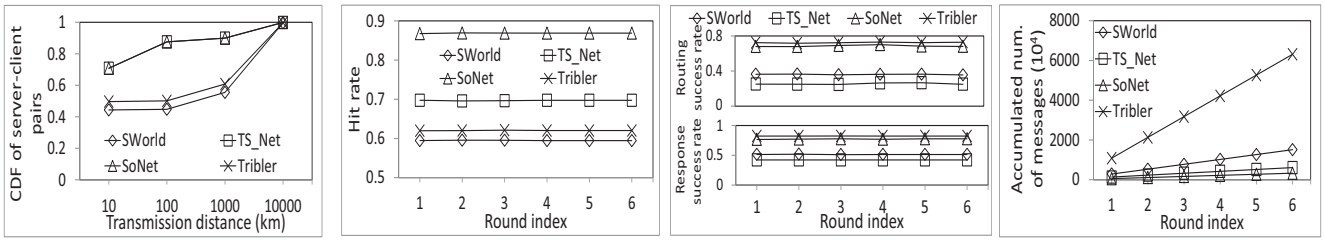
Figure 6(a) shows the CDF of queries versus file search path length in hops. It shows that SoNet has 18.7%, 33.8% and 5.9% more queries resolved within two hops than SWorld, TS_Net and Tribler, respectively. Also, SoNet has fewer queries resolved within long path lengths. Although SWorld clusters common-interest peers as SoNet, and Tribler connects OSN friends as SoNet, SoNet generates shorter path lengths than SWorld and Tribler due to two reasons. First, SoNet has the social based query path selection algorithm to forward queries to the nodes that are likely to resolve the queries. Second, SoNet collects the indices of all files in a subcluster to its head for file querying, so it can always find the file inside the cluster, while SWorld and Tribler have to rely on system-wide lookup DHT function when the intra-cluster search fails. Tribler has more queries resolved in two hops than SWorld and TS_Net, because queries are forwarded using K-multicasting to nodes that are more likely to have the required files than strangers in the same location or having the same interest. SWorld forwards the query between randomly connected peers in an interest cluster that do not have high probability of holding the queried file. TS_Net carries out the file querying along the proximity-aware tree of the requester. Recall that 80% of queries are for files owned by peers within 4 social hop distance of the requester, and the distance between a requester's friends and the requester is usually short. Therefore, a peer has a certain probability to find a queried file from its proximity-aware tree. However, since proximity-close nodes do not necessarily have the same interest, TS_Net produces longer path lengths than other systems that considers either friendship or interest. This figure shows that SoNet generates shorter path length than other methods, which verifies its high searching efficiency.

Figure 6(b) shows the percent of queries resolved in each stage of searching. Inter-cluster stage in SoNet means the *Lookup()* operation to forward the query to the cluster with the queried file's interest if it is not in the requester's cluster. We classified the queries in SoNet that used the *Lookup()* operation to the inter-cluster stage. Inter-cluster stage in other three systems means the complementary system-wide *Lookup()* function. We see that SoNet resolves the highest percent of queries by intra-cluster searching due to its interest and friend clustering features. TS_Net resolves more queries than Tribler and SWorld in intra-cluster searching as it searches all nodes in a location cluster. These results verify the reasons we explained for the different path length performance in Figure 6(a).



(a) The CDF of queries over hops (b) The breakdown of querying (c) The path length in hops (d) The routing distance/latency

Fig. 6: The efficiency of file searching.



(a) CDF of server-client pairs over distance (b) Hit rate within a cluster

Fig. 7: The efficiency of file searching (cont.).

Fig. 8: The trustworthiness of file searching.

Fig. 9: The overhead of structure maintenance.

Figure 6(c) shows the median, 1st percentile and 99th percentile of all path lengths of each of four rounds. We observe that the median and the 99th percentile rates approximately follow SoNet≈Tribler<SWorld<TS_Net. This is due to the same reason as Figure 6(a).

Figure 6(d) shows the median, 1st percentile and 99th percentile of all routing distance and latency, respectively, of each of four rounds. We see that the results follow SoNet<TS_Net<Tribler<SWorld. The routing distance and latency are determined by path length and the distance between hops in the path. From Figure 6(a) and Figure 6(c), we know SoNet generates the shortest path lengths. Also, due to its proximity-aware intra- and inter-subcluster searching, it can resolve the queries by physically nearest servers. Therefore, SoNet produces the least routing distance and latency. Though TS_Net generates longer path lengths than Tribler and SWorld, it generates shorter routing distance due to its proximity-aware searching within a cluster, which reduces its routing distance and latency. The median routing distance and latency of SWorld are slightly longer than Tribler. In Tribler, a peer searches files in its social friends, while in SWorld a peer searches files in its common-interest peers. As most of friend pairs are physically close, Tribler produces shorter median routing distance and latency than SWorld.

Figure 7(a) shows the CDF of server-client pairs over distance, which indicates the efficiency of file transmission from the server to the client. The figure shows that both TS_Net and SoNet have more clients served by servers within shorter distance than other methods. They have 34% and 29% more queries responded by peers within 1000km than SWorld and Tribler, respectively. Recall each file has two copies in the system. The proximity-awareness of SoNet and TS_Net enable them to find the physically closer server to the requester. We also see that Tribler produces slightly more server-client pairs within short distance than SWorld, because friends tend to be physically close to each other, but common-interest peers are scattered over the world. This figure shows the low file transmission latency of SoNet due to its proximity-aware searching.

We regard a cluster in Tribler as social friends within 4 hops. We define hit rate as the percent of queries resolved within a cluster. Figure 7(b) shows the intra-cluster hit rate, which follows SoNet>>TS_Net>>Tribler>SWorld. SoNet has the highest hit rate, because both OSN-based intra-subcluster searching and interest-based intra-cluster searching guarantee that queries can be resolved within a cluster. TS_Net constructs a locality-awareness tree with all nodes in a location, and forwards the query to all nodes through the tree. As explained previously, a peer has a certain probability to find a queried file from its tree because queried files are likely to be in physically close friends. Because TS_Net searches all nodes in the tree while Tribler and SWorld limit their querying within 4 hops in a cluster, TS_Net generates higher hit rate. In Tribler, friends are more likely to have required files compared to strangers with same interests in SWorld. Thus, Tribler has higher hit rate than SWorld. From the result of Tribler, we see only 62% of all queries can be resolved within social network, which implies that without logical overlay, 38% of queries cannot be resolved. This verifies our previous claim that OSN based searching needs logical overlay to enhance the file availability.

B. The Trustworthiness of File Searching

We assumed that a peer is cooperative in forwarding and responding to a query from its friend [19]. The cooperation probability of forwarding or responding to a query between strangers was randomly chosen from 100%, 50% and 10%. The upper figure in Figure 8 shows the average success rate of query routing of each of six successive rounds. In each hop, the forwarder decides whether to deliver or drop the query based on the cooperation probability. Due to the social based routing, Tribler and SoNet have higher query routing success rate than other two methods. We observe that SoNet's success rate is 3%

lower than Tribler. Tribler uses K-multicasting while SoNet uses K paths. Thus, Tribler resolves more queries by social friends than SoNet, leading to a higher routing success rate. If SoNet also employs the K-multicasting method, it would have the similar routing success rate as Tribler. We also see that SWorld generates higher success rate than TS_Net. Recall that peers in SWorld connected to 20 peers while TS_Net connected to 4 peers. Thus, a peer in SWorld has higher probability to communicate with its friend than in TS_Net. The lower figure in Figure 8 shows the average success rate of querying responses in each of six rounds. The success rate of querying response shows the same tendency as success rate of query routing, which verifies the high performance of SoNet in trustworthy file searching by leveraging social based searching.

C. The Overhead of File Searching

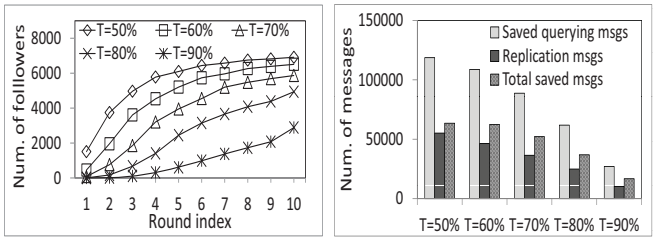
Figure 9 shows the number of messages in system maintenance including those for the maintenance of DHT and subclusters, and the *Insert()* function for file metadata distribution. The structure maintenance is conducted after each round. We see that the system overhead follows SoNet<TS_Net<SWorld<Tribler. SoNet generates fewer messages than other systems for two reasons. First, SoNet does not need *Insert()* function for file metadata distribution. Second, SoNet generates fewer messages for structure maintenance. Tribler maintains all social links, leading to the highest system overhead. SWorld maintains 20 common-interest connections, while in TS_Net, each peer only needs to maintain at most 5 connections to the parent and children. Thus SWorld generated larger number of maintenance messages than TS_Net. The lightest overhead of SoNet indicates its high scalability for millions of users in a file sharing system.

D. Follower based File Replication

Recall that we have file replication strategies for three cases. Here, we use the follower based file replication as an example to show the efficiency enhancement from file replication. In each cluster, we randomly selected a node to be followee, who had 50 files of its interest in initial round. Then we randomly chose one peer instead of all peers in each subcluster to query a randomly chosen file in the followee at the same rate as previous experiments. We ran the experiment for an initial round and subsequent ten successive rounds. In each round, each followee generates a new file, which will be replicated to followers. We varied the threshold of the percent of visited files (T) for follower determination and measured the performance.

Figure 10(a) shows the number of followers with different threshold values over ten successive rounds. It shows that as the number of rounds increases, the number of followers increases. This is because each node visits more files in the followee as the number of rounds increases and then is more likely to be a follower. The figure also shows that there are more followers with lower T in the same round because a lower T enables more nodes to become followers.

Figure 10(b) shows the total number of saved querying messages, the number of file replication messages and their difference (total saved messages) with different T values in the ten successive rounds. We see that the number of saved



(a) The number of followers (b) The effectiveness of replication

Fig. 10: The efficiency of follower based replication.

querying messages and the number of file replication messages decrease when T increases. As T increases, fewer followers and hence fewer replicas are generated, then fewer queries can be resolved locally in requesters, leading to fewer saved querying messages and fewer replication messages. We also see that the number of total saved messages is at least 16860, which means that the follower based replication algorithm can always save cost in file sharing. We observe that $T = 60\%$ and $T = 50\%$ lead to the maximum number of total saved messages, but $T = 60\%$ generates fewer replication messages. This implies that $T = 60\%$ is the optimal threshold value in our experiment settings.

VI. Conclusion

In this paper, we first have analyzed an open public BitTorrent trace data and verified that clustering physically close nodes and common-interest nodes can improve file searching efficiency in a P2P file sharing system. Though recently proposed OSN-based systems use social links for efficient and trustworthy file searching, they cannot provide file location guarantees in a large-scale P2P system. In order to integrate the proximity- and interest-aware clustering and fully utilize OSNs to further enhance the searching efficiency and trustworthiness, we propose SoNet that incorporates four components: a social-integrated DHT, efficient and trustworthy data querying, social based query path selection, and follower and cluster based file replication. SoNet incorporates a hierarchical DHT overlay to cluster common-interest nodes, then further clusters proximity-close nodes into subclusters, and connects these nodes with social links. This social-integrated DHT enables friend intra-subcluster querying and locality- and interest-aware intra-cluster searching, and guarantees file location with the system-wide DHT lookup function. The social based query path selection further enhances the efficiency of intra-subcluster searching. The file replication algorithm reduces the file querying and transmission cost. Through trace-driven experiments on PlanetLab, we prove that SoNet outperforms other systems in file searching efficiency, trustworthiness and system overhead. In our future work, we will investigate how to predict a user's potential file interests by locality, interest and social relationship and use proactive file recommendation and replication to further enhance the searching efficiency and trustworthiness.

Acknowledgements

We would like to thank Dr. Andy Pavlo in Brown University for providing the BitTorrent trace data. This research was

supported in part by U.S. NSF grants OCI-1064230, CNS-1049947, CNS-1156875, CNS-0917056 and CNS-1057530, CNS-1025652, CNS-0938189, CSR-2008826, CSR-2008827, Microsoft Research Faculty Fellowship 8300751, and Oak Ridge Award 4000111689.

REFERENCES

[1] Kazaa Delivers More Than Tunes. http://www.wired.com/.
[2] D. Hughes, G. Coulson, and J. Walkerdine. Free Riding on Gnutella Revisited: The Bell Tolls? *IEEE Dist. Systems Online*, 2005.
[3] S. Genaud and C. Rattanapoka. Large-Scale Experiment of Co-allocation Strategies for Peer-to-Peer Supercomputing in P2P-MPI. In *Proc. of IPDPS*, 2008.
[4] Y. Liu, L. Guo, F. Li, and S. Chen. A Case Study of Traffic Locality in Internet P2P Live Streaming Systems. In *Proc. of ICDCS*, 2009.
[5] H. Shen and K. Hwang. Locality-Preserving Clustering and Discovery of Resources in Wide-Area Distributed Computational Grids. *TC*, 2011.
[6] M. Yang and Y. Yang. An Efficient Hybrid Peer-to-Peer System for Distributed Data Sharing. *TC*, 2010.
[7] G.A. Koenig and L.V. Kale. Optimizing Distributed Application Performance Using Dynamic Grid Topology-Aware Load Balancing. In *Proc. of IPDPS*, 2007.
[8] F. Lehrieder, S. Oechsner, T. Hossfeld, Z. Despotovic, W. Kellerer, and M. Michel. Can P2P-Users Benefit from Locality-Awareness? In *Proc. of P2P*, 2010.
[9] D. R. Choffnes and F. E. Bustamante. Taming the Torrent: A Practical Approach to Reducing Cross-ISP Traffic in P2P Systems. In *Proc. of Sigcomm*, 2008.
[10] S. Seetharaman and M.H. Ammar. Managing Inter-domain Traffic in the Presence of BitTorrent File-Sharing. In *Proc. of Sigmetrics*, 2008.
[11] H. Zhang, Z. Shao, M. Chen, and K. Ramchandran. Optimal Neighbor Selection in BitTorrent-like Peer-to-Peer Networks. In *Proc. of Sigmetrics*, 2011.
[12] A. Iamnitchi, M. Ripeanu, E. Santos-Neto, and I. Foster. The Small World of File Sharing. *TPDS*, 2011.
[13] M. Li, W.-C. Lee, A. Sivasubramaniam, and J. Zhao. SSW: A Small-World-Based Overlay for Peer-to-Peer Search. *TPDS*, 2008.
[14] G. Chen, C. P. Low, and Z. Yang. Enhancing Search Performance in Unstructured P2P Networks Based on Users' Common Interest. *TPDS*, 2008.
[15] K. C. J. Lin, C. P. Wang, C. F. Chou, and L. Golubchik. SocioNet: A Social-Based Multimedia Access System for Unstructured P2P Networks. *TPDS*, 2010.
[16] X. Cheng and J. Liu. NetTube: Exploring Social Networks for Peer-to-Peer Short Video Sharing. In *Proc. of INFOCOM*, 2009.
[17] Y. Li, L. Shou, and K. L. Tan. CYBER: A Community-Based Search Engine. In *Proc. of P2P*, 2008.
[18] H. Shen and C.-Z. Xu. Leveraging a Compound Graph based DHT for Multi-Attribute Range Queries with Performance Analysis. *TC*. 2011.
[19] E. Pennisi. How did Cooperative Behavior Evolve? *Science*, 2005.
[20] J. A. Pouwelse, P. Garbacki, J. Wang, A. Bakker, J. Yang, A. Iosup, D. H. J. Epema, M. Reinders, M. van Steen, and H. J. Sips. Tribler: A Social-based Peer-to-Peer System. In *Proc. of IPTPS*, 2006.
[21] D. N. Kalofonos, Z. Antonious, F. D. Reynolds, M. Van-Kleek, J. Strauss, and P. Wisner. MyNet: A Platform For Secure P2P Personal And Social Networking Services. In *Proc. of PerCom*, 2008.
[22] S. Marti, P. Ganesan, and H. Garcia-Molina. SPROUT: P2P Routing With Social Networks. In *Proc. of P2P&DB*, 2004.
[23] S. Marti, P. Ganesan, and H. G. Molina. DHT Routing Using Social Links. In *Proc. of IPTPS*, 2004.
[24] B. Popescu, B. Crispo, and A. Tanenbaum. Safe and Private Data Sharing With Turtle: Friends Team-Up And Beat The System. In *Proc. of SPW*, 2004.
[25] V. Carchiolo, M. Malgeri, G. Mangioni, and V. Nicosia. An Adaptive Overlay Network Inspired By Social Behavior. *JPDC*, 2010.
[26] Q. Lian, Z. Zhang, M. Yang, B. Y. Zhao, Y. Dai, and X. Li. An Empirical Study of Collusion Behavior in the MAZE P2P File-Sharing System. In *Proc. of ICDCS*, 2007.
[27] PlanetLab. http://www.planet-lab.org/.
[28] C. Wang and X. Li. An Effective P2P Search Scheme to Exploit File Sharing Heterogeneity. *TPDS*, 2007.
[29] S. Seshadri and B. Cooper. Routing Queries through a Peer-to-Peer InfoBeacons Network Using Information Retrieval Techniques. *TPDS*, 2007.
[30] H. Shen, L. Zhao, H. Chandler, J. Stokes, and J. Li. Toward P2P-based Multimedia Sharing in User Generated Contents. In *Proc. of INFOCOM*, 2011.
[31] N. Rammohan, Z. Miklos, and K. Aberer. Towards access control aware p2p data management systems. In *Proc. of the 2nd International workshop on data management in peer-to-peer systems*, 2009.
[32] M. Mcpherson. Birds of a Feather: Homophily in Social Networks. *Annual Review of Sociology*, 2001.
[33] M. P. Wittie, V. Pejovic, L. Deek, K. Almeroth, and B. Y. Zhao. Exploiting Locality of Interest in Online Social Networks. In *Proc. of CoNEXT*, 2010.
[34] Z. Li, H. Shen, H. Wang, G. Liu, and J. Li. SocialTube: P2P-assisted Video Sharing in Online Social Networks. In *Proc. of IEEE INFOCOM Mini-Conference*, 2012.
[35] BitTorrent User Activity Traces. http://www.cs.brown.edu/p̃avlo/torrent/.
[36] N. Laoutaris, D. Carra, and P. Michiardi. Uplink Allocation Beyond Choke/Unchoke. In *Proc. of CoNEXT*, 2008.
[37] R. Zhou and K. Hwang. PowerTrust: A Robust and Scalable Reputation System for Trusted Peer-to-Peer Computing. *TPDS*. 2007.
[38] R. Zhou, K. Huang, and M. Cai. GossipTrust for Fast Reputation Aggregation in Peer-To-Peer Networks. *TKDE*, 2008.
[39] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan. Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications. *TON*, 2003.
[40] A. Rowstron and P. Druschel. Pastry: Scalable, Decentralized Object Location and Routing for Large-scale Peer-to-Peer Systems. In *Proc. of Middleware*, 2001.
[41] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A Scalable Content-Addressable Network. In *Proc. of SIGCOMM*, 2001.
[42] H. Shen, C. Xu, and G. Chen. Cycloid: A Scalable Constant-Degree P2P Overlay Network. *Performance Evaluation*, 2006.
[43] D. Karger, E. Lehman, T. Leighton, M. Levine, D. Lewin, and R. Panigrahy. Consistent Hashing and Random Trees: Distributed Caching Protocols for Relieving Hot Spots on the World Wide Web. In *Proc. of STOC*, 1997.
[44] A. Fast, D. Jensen, and B. N. Levine. Creating Social Networks to Improve Peer-to-Peer Networking. In *Proc. of KDD*, 2005.
[45] A. Iamnitchi, M. Ripeanu, and I. Foster. Small-World File-Sharing Communities. In *Proc. of INFOCOM*, 2004.
[46] L. Backstrom, E. Sun, and C. Marlow. Find Me If You Can: Improving Geographical Prediction with Social and Spatial Proximity. In *Proc. of WWW*, 2010.