

**Interest-Based Self-Organizing Peer-to-Peer Networks:  
A Club Economics Approach**

Atip Asvanund, Ramayya Krishnan, Michael D. Smith, and Rahul Telang

H. John Heinz III School of Public Policy and Management  
Carnegie Mellon University  
Pittsburgh, PA 15213

{atip, rk2x, mds, rtelang}@andrew.cmu.edu

This Version: September 2004

Available from: <http://ssrn.com/abstract=585345>

Acknowledgements: The authors thank Jamie Callan, Jie Lu, and participants at the 2003 Workshop on the Economics of Peer-to-Peer Systems, the 2003 Workshop on Information Technology and Systems, the University of Texas at Austin, the University of Maryland Institute for Advanced Computer Studies, and Carnegie Mellon University's Tepper School of Business for valuable comments on this research. Financial support was provided by the National Science Foundation through grant IIS-0118767.

# **Interest-Based Self-Organizing Peer-to-Peer Networks: A Club Economics Approach**

## **ABSTRACT**

Improving the information retrieval (IR) performance of P2P networks is an important and challenging problem. Recently, the computer science literature has tried to address this problem by improving the efficiency of search algorithms. However, little attention has been paid to improving performance through the design of incentives for encouraging users to “share” content and, mechanisms for enabling peers to form “communities” based on shared interests.

Our work draws on the club goods economics literature and the computer science IR literature to propose a next generation file sharing architecture addressing these issues. Using the popular Gnutella 0.6 architecture as context, we conceptualize a Gnutella ultrapeer and its local network of leaf nodes as a “club” (in economic terms). We specify an IR-based utility model for a peer to determine which clubs to join, for a club to manage its membership, and for a club to determine to which other clubs they should connect.

We simulate the performance of our model using a unique real-world dataset collected from the Gnutella 0.6 network. These simulations show that our club model accomplishes both performance goals. First, peers are self-organized into communities of interest — in our club model peers are 85% more likely to be able to obtain content from their local club than they are in the current Gnutella 0.6 architecture. Second, peers have increased incentives to share content — our model shows that peers who share can increase their recall performance by nearly five times over the performance offered to free-riders. We also show that the benefits provided by our club model outweigh the added protocol overhead imposed on the network, that our results are stronger in larger simulated networks, and that our results are robust to dynamic networks with typical levels of user entry and exit.

## 1. INTRODUCTION

Peer-to-peer (P2P) networks allow a distributed network of users to share computing resources such as processing, storage, or content. File sharing networks based on decentralized P2P computing architectures have gained popularity in recent years with the emergence of applications such as Napster and later Gnutella and KaZaA. However, P2P networks are being adopted in a variety of other environments, including widespread use in corporate and government settings for knowledge management and distributed collaboration. For example, Groove Networks' P2P products have been adopted by the U.S. Department of Defense and global consulting firms for distributed knowledge management applications. With the growth of these applications, there is a need to analyze the efficiency of these networks and whether efficiency can be improved by incorporating user incentives into network design (Krishnan, Smith, and Telang 2003).

To this end, in this manuscript we address the efficiency of current P2P networks as distributed Information Retrieval (IR) systems, and whether their retrieval performance can be improved by incorporating incentives designed to organize peers into interest-based clubs (communities). Our solution addresses two well-known shortcomings of extant P2P architectures, largely unaddressed in the current literature: the inability of peers to identify other peers in the network with similar content interests, and high levels of free-riding among users.

With respect to the first problem, in extant architectures peers establish connections to a location in the network without regard to their own content interests or the interests of other proximate peers. These connections are used for routing queries for content, and thus the queries may be routed inefficiently, rather than following the paths that would maximize query results. This highlights the importance of enabling peers to identify other peers in the network with similar

interests in content or in their ability to satisfy information needs. However, formulating such a method is nontrivial for at least two reasons. First, content in P2P networks is not uniquely labeled, complicating the identification of peers with matching interests or content. Second, there is typically no centralized planner in these networks that can organize peers based on their content or interests. In this paper, we use techniques derived from the club goods economics and IR literatures to develop a self-organizing community formation mechanism that does not require centralized planning or uniquely identified content.

With regard to the second problem, the problem of free riding has been well documented in P2P file sharing networks (Adar et al. 2000; Asvanund et al. 2002). Free-riding occurs when peers consume scarce network resources (e.g., content, storage, bandwidth) without providing resources to the network in return. Free-riding in P2P networks limits network scalability and leads to allocative inefficiencies (Krishnan et al. 2002; Krishnan et al. 2003). Prior work has proposed reducing free-riding in P2P networks through centrally administered systems that control access to the scarce network resources either through pricing or admission control mechanisms (see Krishnan et al 2003 for a review of this literature). However, as noted above, central administration systems are infeasible in many P2P environments. Instead, in this paper we draw on parallels between work in the economics literature on public goods (Hardin 1968; Olson 1965), club goods (Buchanan 1965), and the provision of information goods in P2P networks to develop a solution that provides peers with incentives to share content based on their own self-interest. The approach has the advantage of being readily implementable in today's distributed P2P networks, requiring little extension to the current protocol.

Thus, we propose a next generation P2P file sharing architecture that employs interest-based self-organization of peers to address the two fundamental shortcomings of extant P2P architec-

tures. We use the model of ultrapeers and leaf nodes in the Gnutella 0.6 architecture as the context, and conceptualize an ultrapeer and its network of leaf nodes as a “club” (in the economic sense). We define utility as a measure of a peer’s ability to satisfy another peer’s information needs and we conceptualize the network as a collection of inter-connected clubs operated by ultrapeers who seek to maximize their club’s utility by accepting the right leaf nodes and connecting to the right adjacent clubs. Leaf nodes seek memberships to the right clubs in order to maximize their private utility. Since all peers wish to connect to the right partners, our model imposes an incentive reinforcing structure on the network, which enables the self-organization of peers into clubs that are based on content interests and aversion to free riding.

To measure the effectiveness of our approach, we simulate our model as a computational game in which leaf nodes and ultrapeers jointly attempt to find their optimal club membership. We develop a simulation platform for testing various aspects of our extended P2P protocol. We parameterize our simulation with real world data collected from Gnutella 0.6 between August 31 and September 29, 2002, which includes information for 10,533 unique hosts. The use of real world data is crucial to our evaluation as we rely on the observed correlation between a peer’s information content and queries to motivate our club formation model. In our evaluation, we employ recall, a standard IR performance measure, improves as the network topology evolves in our model. We compare our protocol’s performance to the baseline network (the standard Gnutella 0.6 protocol), to the best performance achievable by the centralized architectures, and to total network load (taking into account added overhead from the revised protocol).

These simulations demonstrate that our club model is able to address both major problems with current hybrid P2P architectures. First, in our club model peers are 85% more likely to be able to obtain content they are interested in from their local club than in current Gnutella 0.6 networks

— suggesting that our club model leads to self-organizing communities of interest. Second, in our club model by sharing peers can receive nearly five times greater recall than free-riding peers can — suggesting that our club model will place strong incentives on peers to share content. Finally, we are able to show that our performance gains outweigh the small additional overhead our model imposes on the network, that our results are robust to dynamic networks with typical levels of entry and exit by peers, and that our results are strengthened in larger networks. This last result is encouraging since our simulations are conducted on 1,000-4,000 node networks, whereas current Gnutella networks include approximately 500,000 peers.

A major contribution of our work is that we combine the efforts in the economics and IR literatures to propose an interest-based clustering mechanism that is not only implementable with little protocol extension, but also has the benefit of addressing free riding. Further, we base our model validation on an intricate simulation platform that simulates the network protocol stack, calibrated with real-world P2P network data. This makes our work an important contrast to most current work in this area that emphasizes only analytical formulations.

The remainder of the paper is organized as follows. Section 2 reviews the literature. Section 3 presents our model. Section 4 discusses the data used in our simulations. Section 5 discusses the simulations and their results. Section 5.4 concludes the paper and presents any further discussions.

## **2. LITERATURE**

Our work relates to the economics, online community, and computer science literatures. With regard to the economics literature, as noted above recent empirical research suggests that free-riding — consuming network resources without providing resources in return — is extremely

common in P2P networks (Adar et al. 2000), and that free-riding worsens in larger networks (Asvanund et al. 2004). High levels of free-riding limit network scalability and lead to inefficiencies from peers who consume the scarce network resources without providing benefits in return to the network in the form of content, storage, or bandwidth (Krishnan et al. 2002a; Krishnan et al. 2002b). These high levels of free-riding are consistent with the predictions of the public goods economics literature in that the private provision of public goods will be socially inefficient in terms of under-provision (a.k.a. free riding) and over-consumption (a.k.a. “the tragedy of the commons”) (Hardin 1968) and that users have more incentives to free-ride in larger networks (Olson 1965). In both cases these inefficiencies arise because individuals only take into account their private utility when making consumption and provision decisions even though these decisions affect the utilities of other members in the network. Additionally, P2P networks share parallels with the club goods literature (Buchanan 1965, Samuelson 1954). P2P networks exhibit characteristics of non-excludability in that access is typically made available to all users of the network (Krishnan et al 2003). In the ideal case, P2P networks also exhibit non-rivalry in demand when a consumer of content becomes a provider of the content, scaling supply to proportionally meet demand (Asvanund et al. 2004). However, in the presence of free riding (when a consumer does not share the files they download), P2P network resources will exhibit rivalry in consumption (i.e., congestion), as in club goods. It is important to note, that P2P networks differ from the traditional club goods model in the economics literature in several respects. First, club members in P2P networks contribute shared resources rather than monetary payments. Second, a consumer of content also becomes its provider. Third, clubs can have inter-club relationships through ultrapeer-to-ultrapeer connections (Sterbetz 1992).

Our work also draws on the online communities literature. This literature analyzes the benefits online groups can provide to each other in the form of ties to a community, social support, and access to community resources (e.g., Kraut and Attewell 1993, Constant, Sproull, and Kiesler 1996). More recently, several papers have analyzed personal motivations for online group participation, concluding that motivations appear to be driven primarily by altruism and reciprocity (Wasko and Faraj 2000, Gu and Jarvenpaa 2003, Subramani and Peddibhotla 2002). The most related study to P2P networks is performed by Butler (2001) who develops a resource-based model of social interaction in online communities and applies this model to data from listserv communities on the Internet.

There have also been emerging efforts attempting to solve some of these issues in P2P networks in the computer science literature. For example, researchers have investigated the use of ultrapeers to reduce traffic load on low bandwidth peers (Kirk 2003); caching to improve the efficiency of content retrieval (Bhattacharjee et al. 2003); and intelligent linkage promotion based on similarity of interests (Sripanidkulchai, Maggs, and Zhang 2001). Most importantly, much attention in this literature has been attributed to Distributed Hash Tables (DHTs) (Ratnasamy et al., 2001; Stoica et al., 2001; Zhao et al., 2001). Nevertheless, DHT design criteria may not be suitable for the real-world deployment because, among other reasons, content in P2P networks are not uniquely identified and the peers are extremely transient (Chawathe et al., 2003). These shortcomings are specifically addressed with the methods proposed in our work.

Finally, there has been a recent emergence of interdisciplinary efforts addressing free-riding leveling P2P networks. The focus has been on providing participants with incentives to provide content and other resources. These efforts include network pricing (Cole, Dodis, and Roughgarden 2003); micro-payment systems (Golle, Leyton-Brown and Mironov 2001); reputation systems

(Lai et al. 2003); and admission control systems (Kung and Wu 2003). Most of these approaches, however, rely on centralized administration, which as we noted above it infeasible in many common P2P implementations. Further, each of these approaches relies on purely analytic models as opposed to empirically validated protocol extensions as applied in this research.

### **3. MODEL**

#### **3.1. Hybrid Architecture Background**

By definition, all P2P networks allow for the bi-lateral sharing of resources distributed among a community of users. P2P architectures vary, however, in regard to how these resources are cataloged within the network (Krishnan, Smith, and Telang 2003). In first generation P2P networks, such as Napster, the network maintained a central catalog of content on the network. When users logged into the network they uploaded a list of the content they were sharing to this central catalog server; and when users wished to locate a file on the network they would search this catalog to determine which peers could provide the content, ultimately accessing the content directly from the identified peer(s). However, while these centralized P2P networks maximize the reach of queries, the central catalog of content also creates a single point-of-failure for the network and potential scalability problems for larger networks (Asvanund et al 2004).

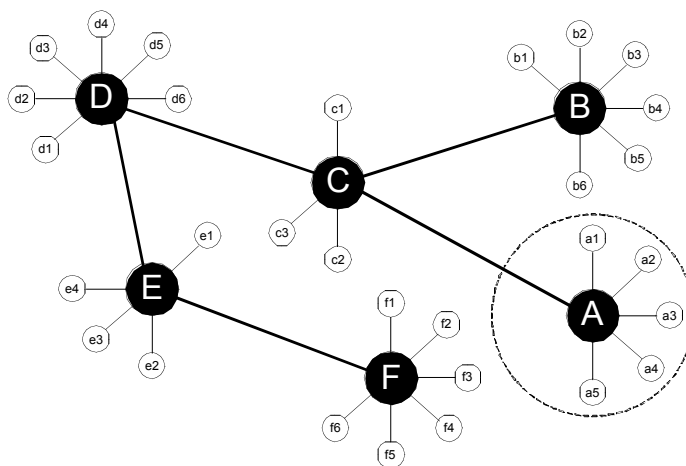
The second generation of P2P networks, such as Gnutella 0.4 (Clip2 2000a), sought to address the weaknesses of the Napster architecture by distributing the catalog throughout the network. Specifically, in Gnutella 0.4, each user on the network maintained a list of the content they were sharing. In the Gnutella 0.4 network users were interconnected in a web, with each peers typically maintaining connections to 3-4 other peers on the network. To make a query, a peer would forward their information request to the 3-4 other peers they were connected to. These peers

would receive this request and reply if they were sharing content matching the query request. They would then forward the query request to each of the 3-4 peers they were connected to, who would perform the same process of responding to and forwarding the query. To prevent the flooding of packets on the network, the protocol only allows query packets to be forwarded a limited number of times (typically 7). This Time-To-Live (TTL) limit, also means that each participant on the network can only reach a limited number of other peers on the network. Thus, the design of these decentralized networks is extremely robust to the failure of individual peers, but at a cost of exacerbating the scalability problems in Napster by placing a high load on peers in processing query requests from their neighbors (Clip2 200b).

Third generation “hybrid” P2P architectures such as Gnutella 0.6 and Kazaa attempted to combine the best features of the first and second generation networks with regard to robust design, and network scalability. Because of this they represent the most popular P2P architecture in use today. In hybrid architecture divides peers on the network into two groups: ultrapeers and leaf nodes. As in centralized P2P networks, leaf nodes connect to ultrapeers and upload a hash of the content they are sharing. Thus, in hybrid networks, ultrapeers maintain a catalog of content shared by its locally connected leaf nodes. However, ultrapeers are also interconnected to each other using a similar protocol to that used in Gnutella 0.4, allowing ultrapeers to forward query requests of adjacent ultrapeers, extending the reach of the network. Finally, ultrapeers only accept a limited number of connections from peers to avoid bottlenecks in updating and querying the catalog that are problematic in centralized networks, and ultrapeers are required to have both large bandwidth and processing capacity to avoid scalability problems in processing queries from neighboring ultrapeers problematic in decentralized networks.

In effect, ultrapeers shield leaf nodes from receiving unnecessary messages that may overwhelm their connections and cause the problems encountered by Gnutella 0.4. Using the file names listed in the content hash, an ultrapeer will forward queries only to the leaf nodes that are likely to have matching content. The content hash of a given peer is essentially a hash table representing the occurrence of the words in the file names of the peer's content. Figure 1 illustrates a Gnutella 0.6 topology. Ultrapeers are depicted by the dark circles, and leaf nodes by the light circles. Ultrapeer interconnectivity is depicted by the thick lines, while leaf nodes' connections to ultrapeers are depicted by the thin lines. As shown by the figure, leaf nodes operate behind ultrapeers, which access the network on their behalf.

**Figure 1: Gnutella 0.6 architecture**



Finally, it is important to note that each of these P2P network architectures share three distinguishing characteristics. First, peers in all three networks exhibit high levels of free riding (Adar and Huberman 2000; Asvanund et al 2004; Krishnan, Smith, and Telang 2003) — consuming network resources without provisioning resources to the network in return. Second, in each case positive and negative network externalities limit network scalability (Asvanund et al 2004). At some point, the marginal costs an additional peer imposes on the network in terms of congestion

on shared resources will outweigh the value they provide to the network in terms of increased content. This limit on scalability is explicitly incorporated into the design of second and third generation architectures through the TTL limit on query forwarding. Third, peers establish connections at a random point in the network. This is particularly critical for second and third generation networks where reach is limited. Each of these three characteristics — low sharing, limited scalability, and lack of content-specific structure — create inefficiencies for the provision of content. The goal of this paper is to use user-level incentives to reduce the impact of these sources of inefficiency.

Because of their popularity and improved scalability, we use the hybrid P2P architectures as the context for this research. Of the hybrid networks, we chose the Gnutella 0.6 protocol as the basis for our research (Kirk 2003) because it is among the most popular P2P networks in use, and has an open protocol.<sup>1</sup> This open protocol allows us to simulate the operations of the network and to obtain representative data on P2P network users. It must be, however, noted that even though our evaluation is performed on the Gnutella architecture and its user data, our models and analyses also apply to any contemporary and future P2P networks that are built on the ultrapeer architecture (e.g., KaZaA, Morpheus). We conceptualize the Gnutella 0.6 network as a collection of clubs, in which an ultrapeer manages a club with its connected leaf nodes as members. Club utility is defined as the sum of the utility of the ultrapeer and its connected leaf nodes. Clubs are operated by ultrapeers who seek to maximize their club utility by accepting the right leaf nodes, while leaf nodes seek memberships to the right clubs in order to maximize their private utility. In addition, ultrapeers also choose to connect their club to the right adjacent clubs to further maximize their club utility. This framework imposes an incentive reinforcing structure on the net-

---

<sup>1</sup> See <http://www.limewire.com/english/content/netsize.shtml>.

work, which encourages peers to form the club efficiently for content discovery, and improves cooperation through sharing.

### 3.2. Information Retrieval Methods

We define the utility an individual peer receives from its neighbors as the degree to which the neighbors are able respond to the peer’s future information needs. We face two problems in operationalizing this measure. First, future queries typically are not known at the time peers need to evaluate the utility provided by other peers on the network and second that content isn’t uniquely identified making it more difficult to determine similarity between two lists of content.<sup>2</sup>

We address the first problem by assuming a peer’s content interests are relatively stable over time, meaning that their current content is a reliable proxy for their future information needs.<sup>3</sup>

We address the second problem using recently developed techniques from the IR literature. These IR methods utilize word frequency statistics for quantifying similarity of interest between catalogs and are designed to function in an unstructured environment such as that found in P2P networks. Specifically, we use the Jensen-Shannon divergence method (Dhillon et al. 2002). Jensen-Shannon divergence produces a scalar value between zero and one, with a lower value signifying higher similarity. To ensure that this measure scales conveniently with our utility model by having a higher value signifying higher similarity, we formulate  $SIM(I, J)$  for  $I, J \in P$ , where  $P$  is the set of all peers (leaf nodes and ultrapeers), as in equation (1). In this equation,  $JS$  specifies Jensen-Shannon divergence, and it quantifies the similarity between two word frequency histograms,  $p_I(V)$  and  $p_J(V)$ , which in our case are the word frequency in either a node’s queries or the names of the content they are sharing.

---

<sup>2</sup> For example, “Britney Spears’s Baby One More Time,” “Baby One More Time: Britney Spears,” or “Britney Spears’s Favorite Hit” may all potentially identify the same content.

<sup>3</sup> This assumption is borne out through a significant correlation between a peer’s content and its future queries.

$$SIM(I,J) = 1 - JS(p_I(V), p_J(V)) \quad (1)$$

In our evaluation, we employ *recall*, a standard IR performance measure, as our primary performance measure. Formally, recall is *the percentage of all the potential matches on the network that are returned for each query*. Our goal is to assess how recall improves as the topology evolves according to our model. In making the performance assessment, we compare the performance improvement over our baseline network: Gnutella 0.6.

Since our model requires peers to exchange some information before making a connection decision (for the application of our similarity measure), it imposes a small amount of additional overhead on the network over and above what would be required in the base protocol. This includes the additional content hash that must be transmitted for the peers to employ our similarity functions. For this reason, our key cost measure is network load: *the amount of traffic transmitted within the network*.

### 3.3. Model

#### 3.3.1. Utility Functions and Information Sets

In this paper, we treat network participants as economic actors and develop a set of incentives such that the participants will choose actions that are both individually rational and improve the overall social welfare for all participants. We also design our model to work in a real world decentralized setting where there is no centralized social planner and peers behave according to their own self-interest. Our model development follows a standard game theoretic specification of each participant's information set, strategy set, and utility function. Following the Gnutella 0.6 architecture, our model participants are leaf nodes and ultrapeers. The aim of our club model is to create self-forming communities of interest where peers with similar content interests are col-

located in the same ultrapeer communities and where peers who share content are better able to join ultrapeer communities with content matching their interests.

**Utility function:** We develop a utility maximizing model for a peer to estimate the benefits of establishing a connection to another peer. In our model we assume that each peer on the network obtains utility  $U$  from other peers on the network which is a function of the likelihood that the other peer will be able to respond to the peer's information needs. As noted above, we operationalize this measure using Jensen-Shannon divergence such that utility that a peer  $p$  receives from another peer  $p'$  is given by

$$U(p, p') = F(SIM(p, p') * K(p')) \quad (2)$$

where  $SIM$  is defined in equation (1) and  $K(p')$  is simply the number of pieces of content shared by  $p'$ .

**Information set:** Each peer on the network has content that can be shared and this content can be summarized in the form of a content hash. In the case of text documents, the content hash could include a word frequency histogram of all of the words in the document. More commonly, this content hash will include a histogram of the word in the metadata describing the content. In most current P2P networks, this metadata is simply the content's filename (which typically includes the recording artist, track number, song title, and album in the case of music files or the show name, episode title, and season and episode number in the case of television shows). However, metadata can also be more descriptive such as ID3 tags in MP3 filenames.<sup>4</sup> Given a content hash for each peer on the network, we propose a modification (described in more detail below) to

---

<sup>4</sup> See [www.id3.org](http://www.id3.org)

the existing Gnutella 0.6 protocol to allow peers to discover the content hash of other peers on the network.

We further assume that the peer information set includes a subset of other peers on the network. This is accomplished in the current Gnutella protocol through each peer's host cache, which keeps track of the IP addresses of other peers in the network that a peer has come in contact with or knows about through PONG messages. A peer can also obtain information about other peers on the network by querying host-cache servers, dedicated servers in the network whose sole purpose is to tell a peer about other peers in the network.

**Strategy set:** In the current Gnutella protocol, a peer's only strategic decision is whether to share their content. Peers also take non-strategic actions to connect to ultrapeers in the network. Specifically, peers do this by selecting an ultrapeer at random from their host cache. Similarly, ultrapeers take non-strategic actions by accepting all requested connections up to a predefined maximum number of connections.

In our model, we modify this strategy set to allow leaf nodes and ultrapeers to make utility maximizing strategic decisions about which connections they will initiate and accept. Leaf nodes decide which ultrapeers to request a connection to maximize the leaf node's utility. In doing this, the leaf node considers utility it receives from the club managed by the ultrapeer — both content shared by the ultrapeer and content from the ultrapeers' immediately connected leaf nodes. Likewise, ultrapeers attempt to choose which leaf node connections they will accept connections from and which other ultrapeers to connect to in a way that maximizes utility in their club.

Specifically, let  $UC(l, up)$  define the utility leaf node  $l$  receives from the club managed by ultrapeer  $up$ :

$$UC(l, up) = U(l, up) + \sum_{l' \in LUP(up)} U(l, l') \quad (3)$$

where  $LUP(up)$  defines the set of all leaf nodes connected to  $up$ . Based on this definition,  $l$  will solve the following maximization problem

$$\max_{\substack{UPL(l) \\ up \in UPL(l)}} \sum UC(l, up) \text{ such that } \|UPL(l)\| \leq MUP(l) \quad (4)$$

where  $MUP: L \rightarrow \{0, 1, \dots\}$  is the maximum number of ultrapeers a leaf node can be connected to, and  $UPL(l)$  is the set of ultrapeers that  $l$  is connected to.

Ultrapeers consider which other ultrapeers to request a connection to, and which leaf nodes and other ultrapeers to accept connections from. As noted above, an ultrapeer considers its club utility, which includes its own private utility and the private utility of the leaf nodes immediately connected to it. An ultrapeer's decisions can significantly affect not only its own utility, but also the utility of its leaf nodes. When considering a leaf node to accept a connection from, the ultrapeer considers the utility that the leaf node will provide to the club. Specifically, let  $UC(up, l)$  define the utility the club managed by  $up$  receives from  $l$

$$UC(up, l) = U(up, l) + \sum_{l' \in LUP(up)} U(l', l) \quad (5)$$

The ultrapeer will then decide which leaf node connections to accept based on the following maximization equation

$$\max_{LUP(up)} \sum_{l \in LUP(up)} UC(up, l) \text{ such that } \|LUP(up)\| \leq ML(up) \quad (6)$$

where  $ML: UP \rightarrow \{0, 1, \dots\}$  is the maximum number of leaf nodes that an ultrapeer can be connected to.

Similarly, ultrapeers accept and request connections to other ultrapeers based on the utility its club receives from the club managed by the other ultrapeer. Formally, let  $UC(up, up')$  define the club utility ultrapeer  $up$  receives from the club managed by  $up'$ :

$$UC(up, up') = \left[ U(up, up') + \sum_{l' \in LUP(up')} U(up, l') \right] + \sum_{l \in LUP(up)} \left[ U(l, up') + \sum_{l' \in LUP(up')} U(l, l') \right] \quad (7)$$

As before, given this utility function, the ultrapeer will choose  $UPUP(up)$ , the set of ultrapeers to connect to, according to the following maximization problem:

$$\max_{UPUP(up)} \sum_{up' \in UPUP(up)} UC(up, up') \text{ such that } \|UPUP(up)\| \leq MUP(up) \quad (8)$$

where  $MUP: UP \rightarrow \{0, 1, \dots\}$  is the maximum number of other ultrapeers an ultrapeer can be connected to, which is defined by the protocol or by the processing capacity of the ultrapeer.

### 3.3.2. Network Evolution Algorithms

Implementing these optimization algorithms within the current Gnutella protocol while minimizing added overhead is complicated by the fact that peers do not have perfect knowledge of all other peers in the network, both parties to a connection must actively establish a protocol handshake (which transmits the content hash) when initiating a connection, and both parties must determine that it is in their self-interest to complete the connection. In this section, we outline the evolution algorithm necessary to incorporate our model into the existing Gnutella protocol.

In the current Gnutella 0.6 protocol, peers request connections by sending a Gnutella connect message to one of the foreign peers in its host cache (Algorithm 1). If the foreign peer can accept this connection it responds with a Gnutella OK message and the connection is initiated by having the connecting peer reply with its content hash.

### Algorithm 1: Existing Gnutella 0.6 Connection Algorithm

Initiating Peer (P) sends <b>Gnutella Connect</b> message <sup>5</sup> to Receiving Peer (P*)	⇒	
	⇐	P* responds with <b>Gnutella OK</b> message <sup>6</sup> if it can accept the connection and any other message if it cannot
P opens connection by sending its Content Hash to P*	⇒	

### Algorithm 2: Recipient's Algorithm

Initiating Peer (P) sends <b>Gnutella Connect</b> message to Receiving Peer (P*) containing P's content hash	⇒	
	⇐	P* responds with <b>Gnutella OK/Connect</b> message, containing P*'s content hash, if P's utility is larger than the lowest utility among P*'s existing connected peers (and any other message otherwise)
P responds with <b>Gnutella OK</b> message if P*'s utility is larger than the lowest utility among P's existing connected peers (and any other message otherwise)	⇒	
If a connection is established, P and P* drop the connected peer with the lowest utility in favor of the new peer		

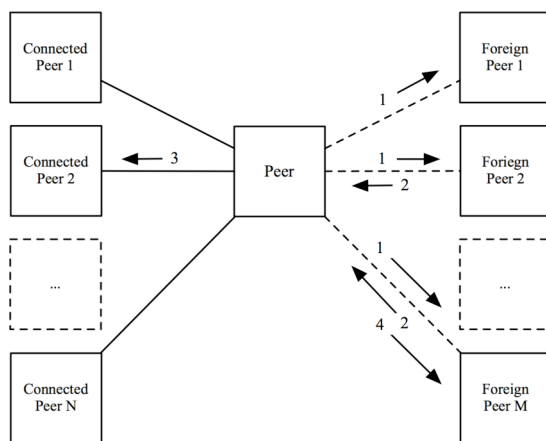
In our modification to this connection algorithm (Algorithm 2), the initiating peer first sends a Gnutella connect message to one of the peers in its host cache, along with the initiating peer's content hash. The responding peer uses the content hash to determine if the initiating peer offers more utility than at least one of the existing connected peers, and if so replies with a Gnutella OK message including its content hash. The initiating peer then uses this hash to determine if the responding peer offers more utility than at least one of its currently connected peers and if so responds with a Gnutella OK message. This sequence of steps is summarized in Figure 2.

<sup>5</sup> GNUTELLA CONNECT/<protocol version string>\n\n. See Kirk (2003) for details of the Gnutella 0.6 protocol.

<sup>6</sup> GNUTELLA/<protocol version string> 200 OK\n\n.

Note that the threat of being dropped and the threat of being refused a connection provide the incentives for peers to behave in a way that is aligned with the network’s interests. Further, since all peers wish to connect to partners offering the highest utility, our model imposes an incentive reinforcing structure on the network, which enables self-organization of peers into communities that are based on content interests and aversion against free riders.

**Figure 2: Club Formation Sequences**



**Note:** 1) connection requests by (originating) peer; 2) replies to connection request from the foreign peers (recipients); 3) originating peer disconnects “Peer 2” to make room for a new peer; 4) originating peer connects to new peer which provides better utility.

Thus, our club formulation can be implemented with the following minor extensions to the existing Gnutella 0.6 protocol: 1) upgrading content hash information, 2) allowing peers to base their connection decision on the content hash they receive, and 3) modifying the handshake protocol to allow both peers to submit their content hash during their connection request. Currently Gnutella 0.6 only stores binary information for each word in the content hash, which specifies whether a peer’s content contains a particular word. To use our club model, the content hash must be upgraded to store word frequencies to be used in the calculation of our utility function. Also, peers must now base their connection decision on the content hash it receives. Finally, in Gnutella 0.6, only the initiating peer submits its content hash, and only after the connection re-

quest has been accepted. For our club model, both peers must be allowed to submit their content hash during the initiation of the connection request. This change will impose additional overhead on the network, and we discuss the implications of this overhead in more detail below.

### ***3.3.3. Simulation Model***

We evaluate the performance of our club formation method using discrete-event simulation, which is performed in two parts: a) query simulation, and b) network evolution simulation. For query simulation, we simulate the query relaying and query answering of the peers for all peers in the network to measure IR performance. For network evolution, we simulate club formation as a series of discrete steps, characterized by a peer's execution of its network evolution algorithm. For each step, a peer (a leaf node or an ultrapeer) is randomly chosen to execute its network evolution algorithm. Each peer's algorithm is executed as an atomic event, during which no other peers, except for the peers that receive connection requests, can perform any actions resulting in changes to the network topology. After the peer completes its algorithm the next peer is randomly chosen to execute its algorithm. If the chosen peer is a leaf node, it evolves its connection to the ultrapeers. An ultrapeer, on the other hand, evolves its connection to other ultrapeers.

We interleave our query simulation and our network evolution simulation in the following manner. We initially simulate query performance in a randomly constructed hybrid network topology — representative of the performance of the (baseline) Gnutella 0.6 network. After this we perform our network evolution algorithms for a number of steps — each step represents a peer being chosen to execute its algorithm. Then we repeat our query performance simulation in order to measure any performance improvement that is achieved. Finally, we measure the overhead costs imposed by our club formation method.

## 4. DATA

To parameterize our simulation with real world data, we collected data from Gnutella 0.6 between August 31, 2002 and September 29, 2002. We modified a Gnutella client to collect users' information and their usage patterns on our behalf. We observed 10,533 unique peers, of which, 8,858 are leaf nodes and 1,675 are ultrapeers (see Table 1 for summary statistics).

**Table 1: Descriptive Statistic of Empirical Data**

Category	Variables	Value	Mean	Std. Dev.	Min.	Max.
<i>Global Count</i>	Node Count	10,533				
	Leaf Node Count	8,858				
	Ultrapeer Count	1,675				
	Nodes With No Content	4,726				
	Nodes With No Queries	6,075				
<i>Individual Nodes</i>	Node Session Length (Secs.)		2,790.73	18,878.57	8.94	990,988.84
	Query Count Per Node		7.73	129.51	0	8,357
	Query Word Count		4.44	4.08	1	39
	Content Count Per Node		96.30	454.40	0	21,206
	Content Name Word Count		5.31	3.87	1	45

Values for global count variables are actual values, while values for individual node variables are means, standard deviations, minimums, and maximums.

Note that 45% (4,726) of the nodes on the network do not provide content. This is consistent with the findings of prior studies that P2P networks exhibit high levels of free-riding, if slightly lower than Adar and Huberman's (2000) finding that 66% of Gnutella 0.4 users were free-riders in August 2000.

## 5. RESULTS

### 5.1. Main Simulation Results

We begin our analysis by simulating a network topology of 2,000 peers, of which 1,800 peers are leaf nodes, and 200 peers are ultrapeers.<sup>7</sup> Each leaf node can be a member of only one club, while each ultrapeer can be connected to three adjacent ultrapeers. The ratio between ultrapeers

---

<sup>7</sup> Note that this represents a fraction of the size of the current Gnutella 0.6 network, which currently numbers nearly 500,000 nodes (<http://www.limewire.com/english/content/netsize.shtml>). Unfortunately, we were unable to simulate the operation of larger network because of the computational demands of our simulation platform.

and leaf nodes is chosen to approximate the actual ratio of ultrapeers to total peers observed in the network Gnutella 0.6 network.<sup>8</sup> In our club formation algorithms, each peer has knowledge of 5% of the foreign peers in the network.<sup>9</sup> For each simulation trial, we evaluate the number of evolutions required to achieve the maximum performance, and assess the marginal protocol overhead imposed on the network and the resulting network. In our notation, an evolution is a period in which, on average, each peer has executed its algorithm once (i.e., an evolution equals 2,000 rounds of community formation algorithms). We then repeat this simulation for a total of 20 trials and report the means and variances of the achieved overhead and recall results.

In Table 2, we show the resulting network performance when we start from a randomly generated topology and perform our algorithms for 10 evolutions.<sup>10</sup> The average recall for our baseline case (a randomly generated topology) is shown using black bars, while the average recall for the optimized case (after 10 evolutions of our algorithms) is shown using white bars. Table 2 further categorizes the results into TTL of 0 (results from the local club), 1, 2, and 3. Because the improvements vary across clubs we separately report performance for all clubs (100%), the top half of all clubs (top 50%), and the top quartile of all clubs (top 25%).

This Table shows that, with the exception of the average recall for TTL=3, our algorithm improves the average recall obtained by all peers. Under our algorithm, peers are on average 85% more likely to find the content they are looking for in their local club (TTL=0) than they are un-

---

<sup>8</sup> On September 2, 2004 ultrapeers represented 9% of total peers in the Limewire Gnutella 0.6 network according to <http://www.limewire.com/english/content/netsize.shtml> (compared to 10% in our simulation). Note that the ratio of ultrapeers to total peers is 16% in our data. This is because we parameterized our data collection client to accept more ultrapeer connections than normal. We did this to ensure that we were able to obtain a sufficient number of both types of peers.

<sup>9</sup> These percentages are experimentally specified. Higher percentages cause the peers to contact more peers during club formation, which may improve the speed of community formation while increasing overhead. Lower percentages would have an opposite effect.

<sup>10</sup> We use 10 evolutions because the performance of the network should have stabilized at this point. Indeed, as will be shown in the next section, the network topology does not gain significant improvement after slightly more than 2 evolutions with our club model.

der the current Gnutella protocol. This ratio reduces to 45% for TTL=1 and 18% for TTL=2 and is finally statistically the same as Gnutella 0.6 for TTL=3. This decline occurs because as the TTL increases, the reach of any individual peer also increases, reducing the difference between the two networks.

**Table 2: Club model performance comparison**

TTL=0				TTL=1			
	Baseline Gnutella	Club Model	Ratio		Baseline Gnutella	Club Model	Ratio
100%	0.0042 (0.0013)	0.0078* (0.0027)	1.85	100%	0.0192 (0.0028)	0.0278* (0.0089)	1.45
50%	0.0061 (0.0016)	0.0160* (0.0057)	2.62	50%	0.0209 (0.0031)	0.0546* (0.0165)	2.61
25%	0.0083 (0.0033)	0.0266* (0.0015)	3.20	25%	0.0222 (0.0039)	0.0880* (0.0259)	3.96
TTL=2				TTL=3			
	Baseline Gnutella	Club Model	Ratio		Baseline Gnutella	Club Model	Ratio
100%	0.0488 (0.0041)	0.0577* (0.0151)	1.18	100%	0.1076 (0.0079)	0.1070 (0.0287)	0.99
50%	0.0494 (0.0061)	0.1111* (0.0230)	2.25	50%	0.1079 (0.0101)	0.1988* (0.0405)	1.84
25%	0.0517 (0.0067)	0.1720* (0.0316)	3.33	25%	0.1086 (0.0124)	0.2959* (0.0488)	2.72

**Note:** Ratio is club recall / Baseline Gnutella recall. Standard deviations shown in parentheses. \* denotes club model recall is statistically significantly different than the baseline Gnutella 0.6 recall.

Our results also show that the top clubs (in terms of recall) see a much stronger increase in recall under our algorithm than under Gnutella 0.6. For example, the quartile of clubs achieve a 220% higher recall from their local club (TTL=0) than Gnutella 0.6 and a 172% higher recall under TTL=3. In investigating this finding further, we observe that this is because under our algorithm, peers who do not share content are unable to join clubs with higher utility and over time become clustered in clubs with other free-riders. Conversely, peers who share desirable content are able to join clubs with other high utility peers. Thus, under our algorithm, a peer who would not have

shared content under Gnutella 0.6, would have the added incentive to share content due to the increased recall they would experience.

We quantify this effect in more detail with an additional experiment. We simulate club formation to compare the value received (as measured by recall) by a peer when the peer provides and does not provide content to the network. We randomly choose a non-free-riding peer and measure the recall it receives in a randomly constructed network.<sup>11</sup> Starting from this random network, we simulate club formation twice. In the first simulation the chosen peer provides its content, and in the second simulation the chosen peer provides no content. Club formation is performed for 10 evolutions in either run. We then compare the recall the peer receives when it provides content and when it does not provide content. We repeat this experiment with 100 randomly chosen peers and display the average of the resulting recall measures in Table 3 for the different time-to-live values.

**Table 3: Change in Recall From Decision to Provision Content**

TTL	Recall		Ratio: Non-Provision/Provision
	Non-provision	Provision	
0	0.0002	0.0112	73.72
1	0.0031	0.0381	12.16
2	0.0109	0.0868	7.97
3	0.0334	0.1585	4.74

The increase in recall across all TTLs is dramatic. Peers who share content receive 74 times higher recall from their local club than peers who free-ride, and receive nearly 5 times higher recall overall.<sup>12</sup> Thus, we would expect that the opportunity to dramatically increase recall would encourage peers who otherwise would not have had sufficient incentives to share content, to

<sup>11</sup> We choose non-free-riding peers for this experiment because otherwise we would not know what content they had available to share when we set the peers to provide content.

<sup>12</sup> These recall results are slightly different than our previously reported recall figures because here we are analyzing recall for only non-free-riding peers who, as noted previously, receive higher recall under our algorithm than free-riding peers.

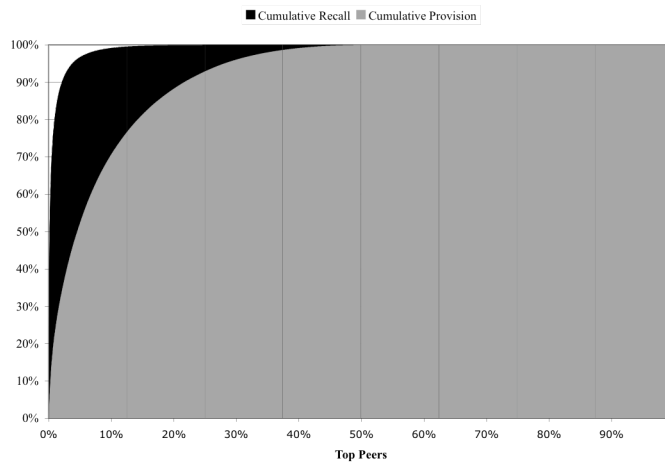
make a choice to share. This in turn would increase the average recall experienced by all peers in the network. Since we do not model this secondary effect, we expect that the change in recall of our algorithm versus Gnutella 0.6, shown in Table 2, is a lower bound on the true change after taking into account reduced free-riding.

We also make two additional important observations from our data. First, while free-riding peers are generally located in the same clubs as other free-riding peers, they are still able to obtain some content from the network, particularly for larger TTL values. This means that a peer who would initially has no content, but is willing to share content, is able to obtain content from the network and over time build up enough content to join high value clubs. Second, as noted above, as we increase TTL, any individual peer has access to a larger segment of the network, reducing the difference between our algorithm and Gnutella 0.6. However, this reduction is particularly acute in our simulation because we only simulate a 2,000 peer structure. In a larger network, such as the current Gnutella 0.6 network which contains approximately 300,000 nodes, even at TTL=3 any individual peer can only reach a small portion of the network. Thus, again we expect our results for larger TTLs represent a lower bound on what would be found in larger networks.

Finally, to analyze information retrieval patterns in the network, we calculate the recall that an individual peer receives from each other peers in the network. We display the cumulative recall and cumulative recall provision for peers in our network sorted in descending order by recall (Figure 3). Cumulative recall depicts the level of recall a peer can receive by having access to the “top” peers. To a given peer, the top peers are those who can provide the highest recall. The Figure thus shows that for any given peer, the top 12.5% peers can provide it nearly 100% recall. This implies that in club formation, if a peer has access to these top 12.5% peers in its club, it will achieve nearly 100% recall. With the topological setup used in this paper (2,000 peers; 10

peers in a club), each peer has access to 0.45% (9) other peers in its club. In the ideal case where for every peer their co-members are the top 0.45% peers, the average recall for all peers is over 58%.

**Figure 3: Characterization of user information retrieval pattern**



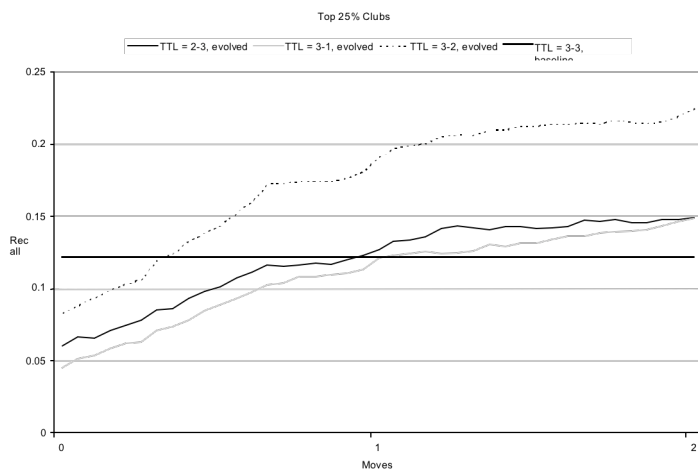
Cumulative recall provision, on the other hand, depicts the level of recall provision “provided” to the network by the top providing peers. The chart shows that the top 20% of the peers provide 85% of the recall to the network. Please note that the top peers in the previous paragraph refer to the top peers to a particular peer, whereas the top peers here refer to the top peers to the entire network. An important implication of this chart is that while every peer in a club can receive the average recall of over 58% if their co-members are the top 0.45% peers (to themselves), many peers in the network share the top peers. Hence, not all communities will have access to the top providing peers. Cumulative recall provision works in opposition with cumulative recall in limiting the potential average recall improvement that can be achieved by club formation. Nevertheless, this is a confirmed property in P2P networks (Adar and Huberman 2000).

## 5.2. Cost Overhead

As noted above, our club formation method imposes additional overhead on the network in terms of transmitting an additional content hash from the receiving to the initiating peer as the network is evolved to an optimized state. In this section, we compare the impact of these “start-up” overhead costs to the longer term benefits to peers from participating in a network with higher recall.

To do this, we take advantage of the fact that under our algorithm, ultrapeers are aware of the content hash offered by each of the other ultrapeers they are connected to. Because of this, under our algorithm ultrapeers should be able to selectively forward query requests to other ultrapeers who are most likely to be able to respond. We use this fact in this section to further reduce the overhead our algorithm imposes on the network.

**Figure 4: Overhead Requirement for Top 25% Clubs**



In Figure 4, we show our results for the top 25% clubs. The straight line labeled (“3-3 baseline”) is the recall that would be achieved under Gnutella 0.6 using TTL=3. The remaining lines make use of the intelligent forwarding feature of our data. Specifically, in this figure, the notation TTL =  $i$ - $j$ , which means that for the setup we use the time-to-live of  $i$ , but we use the best  $j$  (out of 3 total ultrapeer-to-ultrapeer) connections — as determined by interclub utility — from the origi-

nating ultrapeers to relay queries. This table shows that we can achieve. Thus for example, TTL = 3-2 means that we use the time-to-live of 3, but the originating ultrapeer only chooses the best 2 connections out of the 3 connections to broadcast its queries.

This Figure shows that for TTL=2-3, TTL=3-1, and TTL=3-2, we can achieve higher recall using our algorithm and intelligent forwarding than what is achieved by Gnutella 0.6.<sup>13</sup>

We use this fact to demonstrate that the fixed overhead in performing our club model can pay for itself in reducing the variable costs incurred by relaying queries. For example, consider the top 25% clubs. Evolving a network setup with TTL of 2 by one evolution can match the performance of Gnutella 0.6 with TTL of 3. This evolution would require 1,800 steps of leaf node algorithm and 200 steps of ultrapeer algorithm. The most significant cost of our model is the bandwidth required for the transmission of information sets for the peers to make decision about which other peers to connect to. In our formulation, we transmit information sets in the form of compressed word frequency list. From our empirical analysis, we find that a word frequency list of a leaf node when compressed with the popular ZLIB compression algorithm (Deutsch 1996) requires 620 bytes on average for the peers in our network. Likewise, for a club of 10 peers, we find that a word frequency list compressed using the same algorithm requires 5,085 bytes on average. Please note however that we choose a most common compression algorithm, and we do not employ any domain specific techniques for representing a word frequency list (e.g., dictionary, stop words). If specialized domain knowledge were used to improve the compression efficiency, one could obtain even lower overhead.

---

<sup>13</sup> Note that, as above, these recall figure do not take into account any additional gains in recall from reduced free-riding due to the added incentives present in our algorithm.

To calculate how frequently these messages will need to be sent to obtain our results, we observe that in a leaf node algorithm, a leaf node sends its information to 10 ultrapeers each time it evolves its position in the network. Therefore after 1,800 steps our enhancements would have incurred the leaf node overhead cost of  $1,800 * 10 * 620 = 11,160,000$  bytes. On the other hand, for the ultrapeer algorithm, an ultrapeer will contact 10 other ultrapeers in each evolution. Thus in 200 steps, our enhancements will have incurred an overhead cost of  $200 * 10 * 5,085 = 10,170,000$  bytes, yielding the total overhead cost of  $11,160,000 + 10,170,000 = 21,330,000$  bytes.

We can then compare this overhead cost to the savings imposed by intelligently forwarding query packets (instead of sending the queries to all connected ultrapeers). Specifically, for TTL=3, each query message is relayed 27 times in our simulation. According to our data, on average, each query message takes up 46 bytes. Thus, each query message imposes 1,242 bytes of overhead the network. For TTL=2, on the other hand, each query message is relayed 9 times, thus each query message imposes 243 bytes. We can now calculate the number of queries using our optimized network with TTL=2 before the savings from reduced forwarding are larger than the added protocol overhead from our optimization algorithm. Letting  $x$  be the number of query messages that must be sent for the cost overhead of our model to be justified by the reduction in time-to-live,  $x$  is determined by  $21,330,000 < 1,242 * x - 243 * x$ . Thus, 21,351 or more queries must be relayed in the network to justify the overhead cost. This translates to each peer having to make about 11 queries before the cost is justified. Based on our data, each peer issues the average of 10 queries per hour. Therefore, our cost is justified in little over one hour, and the network will reap additive surpluses thereafter.

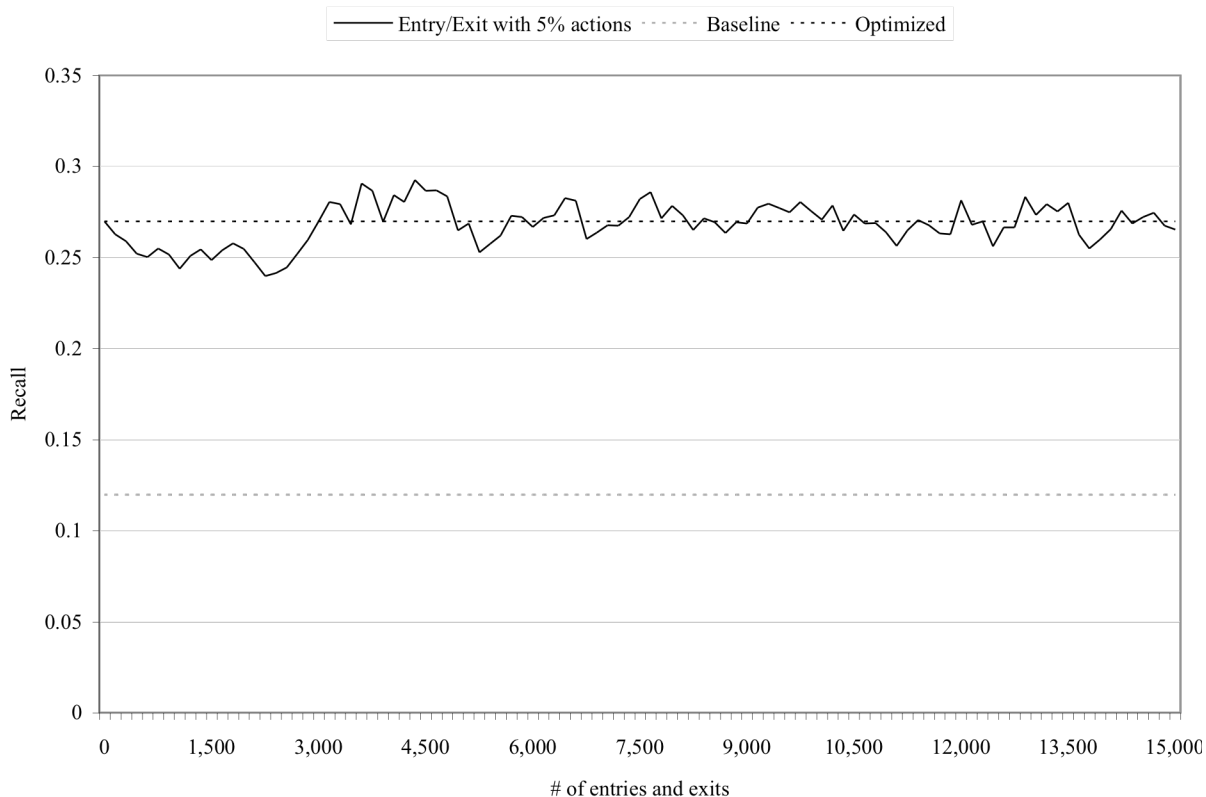
### 5.3. Dynamic Networks

A related question is how robust our results are to entries to and exits from the network by peers. Under such a dynamic network, the cost in optimizing the network is no longer a one-time fixed cost. Rather, our club model must be periodically executed to keep up with the impact that the entries and exits of the peers may have on the network — namely to undo the optimization of our club model. We parameterize our dynamic network simulation using an earlier peer-to-peer study (Asvanund et al. 2002), a user enters and exits the network at the rate of 2 times per day. This translates to roughly 0.083 times per hour. To map this factor into our framework, 8.3% of the active leaf nodes (i.e., leaf nodes currently connected to our network) will randomly exit the network in a given hour, and will be replaced by the same number of new leaf nodes (i.e., leaf nodes in our data sample, but currently not connected to our network), who will be entering the network.

Using this rate of entry and exit, we find that in order to maintain our network in its optimized state, each of the incoming peers (150 peers) must perform our club algorithm, and 5% of the existing peers (100 peers) must also perform our club algorithm. Through experimentation, we found 5% to be the smallest percentage that allows the network to maintain its operation at the optimized level. Figure 5, our recall results in a network setup with 2,000 active peers, with 1,800 leaf nodes and 200 peers are ultra peers, as above. In this setup, 150 randomly chosen active leaf nodes (8.3% of the active leaf nodes) will exit the network, and 150 randomly chosen new leaf nodes join the network each hour after the network has been fully optimized with our club model, as outlined above. As shown in the figure, by having 5% of the existing peers per-

form our club algorithm (in addition to each of the entering peers) we are able to maintain the level of recall achieved in the optimized network without enter and exit.

**Figure 5: Network dynamics when existing peers also perform intelligence**



We can further characterize the additional overhead necessary to optimize this dynamic network. To do this we note that, as above, for each incoming peer to perform their intelligence would consume the total of 837,000 bytes per hour.<sup>14</sup> To have 5% of existing peers perform their intelligence would require additional 1,066,500 bytes per hours, for a total of 1,903,500 bytes per hour. Following section 5.2, the cost savings from using TTL=2 instead of TTL=3 with 10 queries per hour per peer results in 19,980,000 bytes per hours. Thus, in each given hour, the benefit of maintaining the network in an optimized state is roughly 10 times the cost in maintaining the

<sup>14</sup> I.e., 150 new peers per hour \* 5,085 bytes per optimization.

network at its optimal stage, demonstrating the effectiveness of our model can be effective in a dynamic environment.

#### 5.4. Sensitivity Analysis

We perform sensitivity analyses on our results in two ways. First, by analyzing how our results change for smaller and larger networks and second by varying the rate of entries and exits in our dynamic network.

To analyze how our results vary for smaller and larger network we reanalyze our results for networks with 1,000 and 4,000 nodes. In each case, we maintain the number of leaf-to-ultrapeer and ultrapeer-to-ultrapeer connection used in our initial simulation and the same ratio of ultrapeers to total peers.

**Table 4: Club model performance comparison (1,000 Node Network)**

TTL=0				TTL=1			
	Baseline Gnutella	Club Model	Ratio		Baseline Gnutella	Club Model	Ratio
100%	0.0074 (0.0014)	0.0174* (0.0031)	2.35	100%	0.0333 (0.0039)	0.0580* (0.0131)	1.74
50%	0.0100 (0.0025)	0.0318* (0.0074)	3.18	50%	0.0339 (0.0045)	0.1020* (0.0193)	3.01
25%	0.0134 (0.0048)	0.0520* (0.0021)	3.88	25%	0.0428 (0.0041)	0.1582* (0.0321)	3.70
TTL=2				TTL=3			
	Baseline Gnutella	Club Model	Ratio		Baseline Gnutella	Club Model	Ratio
100%	0.0917 (0.0062)	0.1021* (0.0156)	1.11	100%	0.1979 (0.0121)	0.1972 (0.0245)	1.00
50%	0.0923 (0.0076)	0.1716* (0.0261)	1.86	50%	0.1983 (0.0142)	0.2863* (0.0452)	1.44
25%	0.0996 (0.0067)	0.2445* (0.0414)	2.45	25%	0.2199 (0.0162)	0.3631* (0.0451)	1.65

**Note:** Ratio is club recall / Baseline Gnutella recall. Standard deviations shown in parentheses. \* denotes club model recall is statistically significantly different than the baseline Gnutella 0.6 recall.

Our results for a 1,000 node network are very similar to our previous results for a 2,000 node network. As before, peers appear to cluster into groups with similar interests (in our club model peers are 135% of more likely to find the content they are interested from a member of their local club than in Gnutella 0.6). Likewise, the top clubs provide substantially higher recall than other clubs, providing added incentives for peers to share their content.

**Table 5: Club model performance comparison (4,000 Node Network)**

TTL=0				TTL=1			
	Baseline Gnutella	Club Model	Ratio		Baseline Gnutella	Club Model	Ratio
100%	0.0018 (0.0008)	0.0159* (0.0011)	8.83	100%	0.0083 (0.0012)	0.0412* (0.0054)	4.96
50%	0.0023 (0.0014)	0.0239* (0.0034)	10.39	50%	0.0085 (0.0032)	0.0631* (0.0152)	7.42
25%	0.0031 (0.0025)	0.0331* (0.0024)	10.68	25%	0.0095 (0.0024)	0.0892* (0.0254)	9.39
TTL=2				TTL=3			
	Baseline Gnutella	Club Model	Ratio		Baseline Gnutella	Club Model	Ratio
100%	0.0240 (0.0035)	0.0841* (0.0124)	3.50	100%	0.0527 (0.0041)	0.1444* (0.0352)	2.74
50%	0.0257 (0.0064)	0.1279* (0.0325)	4.98	50%	0.0580 (0.0030)	0.2134* (0.0215)	3.68
25%	0.0259 (0.0051)	0.1792* (0.0351)	6.92	25%	0.0596 (0.0045)	0.2944* (0.0336)	4.94

**Note:** Ratio is club recall / Baseline Gnutella recall. Standard deviations shown in parentheses. \* denotes club model recall is statistically significantly different than the baseline Gnutella 0.6 recall.

Our results for the 4,000 are even more dramatic. Peers are nearly 9 times more likely to find the content they are looking for in their local club under our club model than under Gnutella 0.6, our club model provides statistically significantly higher recall for all clubs and all TTL levels (even TTL=3, unlike Tables 2 and 4). Finally, the top quartile clubs provide nearly twice the recall than the average club does. The fact that our results are enhanced in larger clubs is particularly encouraging given that current Gnutella 0.6 networks have approximately 500,000 nodes compared to the 4,000 used in our largest simulation.

Finally, we perform a sensitivity analysis on our dynamic network setup by varying the rate of entries and exits of the peers in a 2,000 node network. We performed the same experiment with a network in which 300 (15%) and 450 (23%) peers enter and exit the network in each given hour. When 300 peers enter the network per hour we find that to maintain the optimal topology, in addition to the incoming peers performing our club algorithms, 15% of the existing peers are required to perform our club algorithms. Similarly for 450 entries and exits per hour, we find that the incoming peers and 35% of the existing members must perform our club algorithms. Based on the previous argument, both of these setups are still cost effective when compared to the benefits provided by the optimized network. However, the increase in the number of existing members required to maintain the optimized network at a stable recall level suggests that as entries and exits increase, they will reach a point where our club model can no longer maintain an optimized network. However, from these simulations this rate of entries and exits appears far above the rate observed in current Gnutella networks.

## **6. DISCUSSION**

P2P networks have gained significant popularity for consumer file sharing and are gaining popularity in corporate and government settings for enterprise knowledge management. However, current P2P networks exhibit two well-known inefficiencies. First, users do not have strong incentives to share content and as a result level of file sharing in P2P networks are below socially optimal levels (Krishnan et al 2002a). Second, network reach is limited (Asvaund et al 2004) and networks are organized without regard to content interest, meaning that in many cases peers are unable to locate their desired content on the network. However, while these two problems are well-known there has been very little research conducted to address these inefficiencies using the

economic characteristics of P2P networks. We use simulations seeded with real-world data to bridge this gap in the literature.

These simulations demonstrate that our club model can result in significantly more efficient file sharing architectures. We do this by combining concepts from the club goods economics literature with concepts from the IR computer science literature. The resulting club model incorporates economic incentives and content-based measures of similarity of interests into the Gnutella 0.6 protocol to create increased incentives for users to share content and create self-forming communities of interest among users. Our models further show that our network improvements are robust after taking into account the possibility of entry and exit by peers, the added protocol overhead imposed by our models, and networks of varying size. Moreover, we find that our results are strengthened in larger networks, which is particularly important given that our simulations are conducted on 1,000 to 4,000 peer networks while the current Gnutella 0.6 network numbers approximately 500,000 peers.

It is important to note that our simulations represent a lower bound on the true gain we expect from incorporating our club models into hybrid P2P networks. First, we do not model increased recall from the increased sharing we would expect due to our incentive structures. As more peers share their content, there will be more content available on the network improving both first-order recall and the ability of our club model to assign peers to the appropriate community of interest. Second, our overhead calculations represent a worst-case scenario. It would be useful to develop utility models of peers' sharing decisions and incorporate this effect into future simulation studies of these club models.

Additionally, with regard to overhead, there are a variety of techniques that can improve cost overhead (e.g., pruning the content hash, improving the host-cache servers), further reducing the cost requirement for maintaining our network. Host-cache servers allow peers to more easily obtain IP addresses of other peers. In the current networks, host-cache servers randomly distribute IP addresses to peers. An extension to the protocol can be made to allow host-cache servers to selectively distribute IP addresses, using a similar utility function to what we define in this work, with the intention to introduce similar peers to each other. This should shorten the number of evolutions for the peers to reach their optimal performance. Content models, on the other hand, can be pruned by removing less frequent words (e.g., words with at most one occurrence) (Lu and Callan, 2003). This can reduce the size of content models by half, while slightly reducing the accuracy of the content models.

Our models could also be extended by allowing peers to join multiple ultrapeers, as is the case in current Gnutella 0.6 networks. Incorporating this feature into our club model may provide improved service to peers who have multiple distinct content interests, possibly further increasing our recall gains.

Finally, a limitation of our model is that we assume all peers to be telling the truth (i.e., reporting the true content hash). A solution to the limitation would be an integration of reputation systems (Lai et al. 2003), which is a large area of study in distributed systems. However, most of the reputation works on P2P networks are based on analytical models rather than empirical approaches. Here again, it would be useful to incorporate these analytic models into our simulation framework to further validate our approach.

## 7. BIBLIOGRAPHY

- Adar, E. and Huberman, B. A. 2000. Free riding on Gnutella. *First Monday* 5(10).
- Asvanund, A., Clay, K., Krishnan, R. and Smith, M. 2002. An Empirical Analysis of Network Externalities in P2P Music-Sharing Networks. ICIS 02, Barcelona, Spain, Dec 15-19.
- Ba, S., J. Stallaert, A.B. Whinston. 2001. Optimal Investment in Knowledge Within a Firm Using a Market Mechanism. *Management Science* 47(9) 1203-1219.
- Bhattacharjee, B., S. Chawathe, V. Gopalakrishnan, P. Keleher, B. Silaghi. 2003. Efficient Peer-To-Peer Searches Using Result-Caching. Proceedings of ITPS2003.
- Buchanan, J. 1965. An Economic Theory of Clubs. *Economica*, New Series, Vol. 32, No. 125. (Feb., 1965), pp. 1-14.
- Chawathe, Y., Ratnasamy, S., Breslau, L., Lanham, N., and Shenker, S. 2003. Making gnutella-like p2p systems scalable. In Proc. of SIGCOMM (2003).
- Clip2, 2000a. The gnutella protocol specification v0.4. Available at <http://www.clip2.com/GnutellaProtocol04.pdf> (January 1, 2001).
- Clip2, 2000b. Gnutella: To the Bandwidth Barrier and Beyond. Available at <http://dss.clip2.com> (Nov. 6, 2000).
- Cover, T. M. and Thomas, J. A. *Elements of Information Theory*, Wiley, 1991.
- Deutsch, L.P. 1996. GZIP Compressed Data Format Specification. Available at <ftp://ftp.uu.net/pub/archiving/zip/doc> (Sep. 11, 2003).
- Dhillon, I., Kumar R. and Manella, S. Information Theoretic Feature Clustering for Text Classification. TextML-2002.
- Golle, P., K. Leyton-Brown, I. Mironov. 2001. Incentive for Sharing in Peer-to-Peer Networks. Working Paper, Computer Science Department, Stanford University, Palo Alto, CA.
- Hardin, G. 1968. The Tragedy of the Commons. *Science*, 162 1243-48.
- Kirk, P. 2003. Gnutella Protocol Development — An Overview of 0.6. Protocol Specification. Available at <http://rfc-gnutella.sourceforge.net/rfcGnutella/index.html> (July 5, 2003).
- Krishnan, R., Smith, M., Tang, Z. and Telang, R. 2002a. The Virtual Commons: Why Free riding Can Be Tolerated in File Sharing Networks. ICIS 02, Barcelona, Spain, Dec 15-19.
- Krishnan R., Smith, M., and Telang, R. 2003. The Economics of Peer-To-Peer Networks. *Journal of Information Technology Theory and Applications*, Forthcoming.

- Kung, H. T., C. Wu. 2003. Differentiated Admission for Peer-to-Peer Systems: Incentivizing Peers to Contribute Their Resources. Working Paper, Harvard University, Cambridge, MA.
- Lai, K., M. Feldman, I. Stoica, J. Chuang. 2003. Incentives for Cooperation in Peer-to-Peer Networks. Working Paper, University of California at Berkeley, Berkeley, CA.
- Olson, M. 1965. *The Logic of Collective Action*. Harvard University Press, Cambridge, MA.
- Rosenthal, D., Roussopoulos, M., Maniatis, P., Baker, M. Economic Measures to Resist Attacks on a Peer-to-Peer Network. Workshop on Economics of Peer-to-Peer Systems, 2003.
- Samuelson, P. 1954. The Pure Theory of Public Goods. *The Review of Economics and Statistics* 36(4) 387-389.
- Sripanidkulchai, K., Maggs, B. and Zhang, H. Efficient Content Location and Retrieval in Peer-to-Peer Systems by Exploiting Locality in Interests. SIGCOMM'01, 2001.
- Sterbetz, F. P. and Sandler T. 1992. Sharing Among Clubs: A Club of Club Theory. *Oxford Economic Papers*, 44(1), 1-19.
- Stoica, I., R. Morris, D. Karger, M.F. Kaashoek, H. Balakrishnan. 2001. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. Proceedings of SIGCOMM'01.
- Zhao, B., KubiaTowicz, J., Joseph, A. 2001. Tapestry: An Infrastructure Fault-tolerant Wide-area Location and Routing. Technical Report. Berkeley, CA, University of California, Berkele