Incentive Mechanisms for Peer-to-Peer Systems

Gitika Aggarwal
Sharath J. George

December 9, 2005

Abstract

Peer-to-Peer (P2P) systems allow participants to share their computational, storage, and networking resources to the benefit of every participant. Most P2P systems assume that all participants in the system follow the protocols and observe the system's fair use policies. However, in a system with open or loosely controlled membership, participants have a self-interest in modifying their behaviour if it allows them to consume the network's resources without contributing any of their own. As the number of these *Free Riders* in the P2P network increases, it not only decreases the effective performance of the network, but also decreases the incentive for others to participate. This brings in the need for good incentive mechanisms to motivate the peers to participate and penalize them if they dont participate. Gnutella and Torrents are two popular P2P network protocols. In this paper we examine several incentive mechanisms that could help improve the *Free Rider* problem in these networks.

1 Introduction

Peer-to-peer (P2P) systems allow participants to share their computational, storage, and networking resources to the benefit of every participant. This cooperative sharing gives participants access to an abundance of resources they could not afford individually. It also enables organic scaling as the system evolves, while requiring no dedicated infrastructure beyond network connectivity.

Most existing P2P systems are designed to address issues such as scalability, load-balancing and fault-tolerance. However, many systems assume that all participants in the system follow the protocols and observe the system's fair use policies. However, in a system with open or loosely controlled membership, participants have a self-interest in modifying their behavior if it allows them to consume the network's resources without contributing any of their own. This phenomenon of selfish individuals who opt out of a voluntary contribution to a group's common welfare has been widely studied, and is known as the *free-rider* problem.

2 Free-Riding in P2P Systems

There are several fronts on which a participant can attempt to *free-ride* the system's resources. Several forms of *free-riding* are summarized below:

P2P file-sharing systems provide decentralized file storage to allow users to download files directly from one another. The work of serving files in virtually all current P2P systems is performed for free by the systems' users. Since users do not benefit from serving files to others, many users decline to return the favor. Thus, these free-riders refuse to contribute their own resources (disk space and network bandwidth) to the system. In fact, two recent studies of the Gnutella network have found that a very large proportion of its users contribute nothing to the system.

To ensure the scalability and robustness of the system, as well as to avoid some legal issues, a centralized database of content of each peer usually does not exist in a P2P system. Instead a distributed catalog of content is favored in which each peer only maintains a list of resources/services of their own, and maybe also of their acquaintances and neighbors. In this distributive model peer discovery is realized via message relaying between peers so that a message for resource searching is propagated in the system with a *word-of-mouth* effect. However, a peer may simply drop a message that is sent from other peers for relaying, for the purpose of saving communication bandwidth and energy. Therefore a message relaying P2P system is vulnerable to the free riding problem. Since a sound P2P system relies on the contribution of resources from each individual peers, free riding can cause severe degradation of the system performance or even paralyze the system.

Unless the P2P system implements a mechanism to reward contributors or penalize defectors, the equilibrium strategy would always be to defect or *free-ride*, thus degrading the system's performance and also demoralizing the contributors. An effective strategy to counter this problem would be to implement an incentive mechanism so that each participant is not expected to contribute to the network altruistically, but to increase the utility that he is premitted to derive from the network. Hence, the incentive mechanism should be such that the equilibrium strategy for each peer would be to contribute to the network and in the event of defection, would cause degradation or no service to the defector.

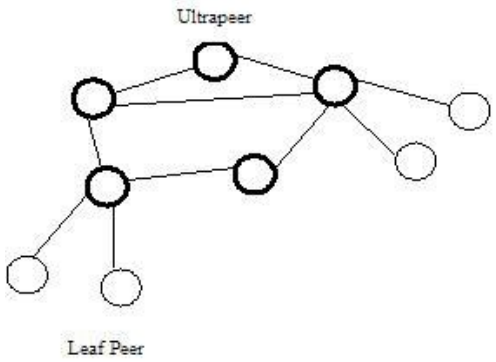

Figure 1: Architecture of Gnutella

3 Gnutella 0.6

3.1 Architecture

The peers in a Gnutella network [1] are classified as "leaf peers" and "Ultrapeers". Peers which have a higher bandwidth and processing power can choose to be ultrapeers. A hierarchy is imposed on the network. Ultrapeers exist at the first tier and each ultrapeer is connected to one or more other ultrapeers. Below this tier is the tier of leaf peers. Each leaf peer connects to one or more ultrapeers. Leaf peers don't establish connections with other leaf peers. An ultrapeer acts as a proxy to the Gnutella network for the leaf peers connected to it by replying to queries on behalf of the leaf peers. This reduces the number of nodes on the network involved in message handling and routing, and the actual traffic among them.

3.1.1 Peer Discovery

Peer discovery in a Gnutella network doesn't depend on a centralized server. Peers are able to locate other peers on the network through the network itself. When a peer needs to discover the addresses of other peers on the network, it sends a PING message to all the nodes that are connected to it. In turn, whenever a peer receives a PING message it replies with a PONG message and then propagates the PING message it received to the other peers that are connected to it. Similarly PONG messages are also relayed through the network and are relayed back to the source which sent the PING message. So a peer who sent PING messages receives the addresses of other peers from the network from all the PONG messages that return to it. Once it has a list of peers, it can choose which peers it wants to connect to.

3.1.2 Searching and Resource Location

A peer issues a query by passing the query string to each of the peers it is connected to, through a QUERY message. When a peer receives the QUERY message, it passes it on to all the other peers it is connected to, except to the peer from which it received the query. Besides passing on the query to other peers, each peer checks if the query string matches with any of the resources it is sharing. If there is a match then it passes a QUERY-HIT message back to the peer from which it received the QUERY. The QUERY-HIT message is transmitted back all the way to the peer which originated the query.

3.1.3 Data Transfer

Once the source peer receives all the QUERY-HIT messages it then selects one or more peers from among the ones that replied and initiates a direct connection with those peers for the data transfer.

4 Incentive Mechanisms for Gnutella

Since there is no centralized control in a Gnutella network, the incentive mechanism has to be enforced in a distributed fashion by implementing it in each node.

4.1 Intelligent Club Management

In this incentive mechanism, the Gnutella network is conceptualized as a collection of clubs [5] operated by ultrapeers who seek to maximize their club value, while the leaf node connects to the right clubs in order to maximize their private utility.

4.1.1 Leaf peer strategy

Let S be the set of all ultrapeers that a leaf node l is connected to. The utility of l is calculated by:

$$U(l) = \sum_{(X|X \in S)} V(X)$$

$V(X)$ is the utility provided to a leaf peer l by an ultrapeer X that l is connected to.

$$V(X) = \sum_{l' \in L | l' \neq l} \alpha * v(l') - \beta * c(l')$$

Where L is the set of all leaf nodes that the ultrapeer X is connected to.

$v(l')$ is the value gained by l from l' and $c(l')$ is the cost incurred by l because of l'. The value contribution of l' $v(l')$, to l is evaluated as a function of the number of bytes shared by l' (sb), the number of files shared by l' (sf), the

number of query hits returned from l' (qh) , the bandwidth provided by l' (b), and the distance between l and l' (d).

$$v(l') = f(sb, sf, qh, b, d)$$

A leaf assigns a higher utility value to leaf nodes which share more content, have higher bandwidth and are geographically closer. The cost $c(l')$ is a function of the number of queries that l' generates (q), and the number of bytes that it downloads (bd).

$$c(l') = f(q, bd)$$

After some interval of time t, the leaf node recalculates the utility value of each ultrapeers that it is connected to. It also discovers a new set of ultrapeers (D).If any ultrapeer in $d \in D$ has a higher utility than an ultrapeer $X \in S$, then l disconnects from X and connects to d.

4.1.2 Ultrapeer Strategy

The utility of an ultrapeer is the weighted sum of the expected utilities provided by each of the leaf nodes that it is connected to. The weights can be decided based on the bandwidth of the leaf node and/or its distance from the ultrapeer. Each ultrapeer knows the set of shared content for each of its leaf peers, and also the other attributes (mentioned in the previous section) that each of the leaf peers use to calculate their individual utility. This enables the ultrapeer to calculate the value and cost each child leaf attributes to the other leaf peers connected to that ultrapeer.

An ultrapeer can accept only limited number of connections from leaf peers. When an ultrapeer reaches capacity and recieves an incoming connection request from a new leaf peer A, it checks if there exists a leaf peer B connected to it that provides lesser utility than A. If yes, then it disconnects B and accepts the connection request from A. This strategy causes the *free-riders* to get disconnected by the ultrapeers often and hence provides incentive to all the leaf nodes to contribute to the network.

4.2 Selfish Link-based InCentive Mechanism (SLIC)

Gnutella uses the flooding-based search mechanism, i.e., when a node wants to find a particular piece of data, it sends a search query to the peers that it is connected to on the P2P network and these peers in turn forward this query to the peers that they are connected to. If any of these peers refuse to forward the query, then the potential search space of the querying peer could reduce quite significantly (depending on the number of peers that it is connected to). Hence, in Gnutella, the flooding based search method enables neighboring peers to control each others' access to the rest of the network.

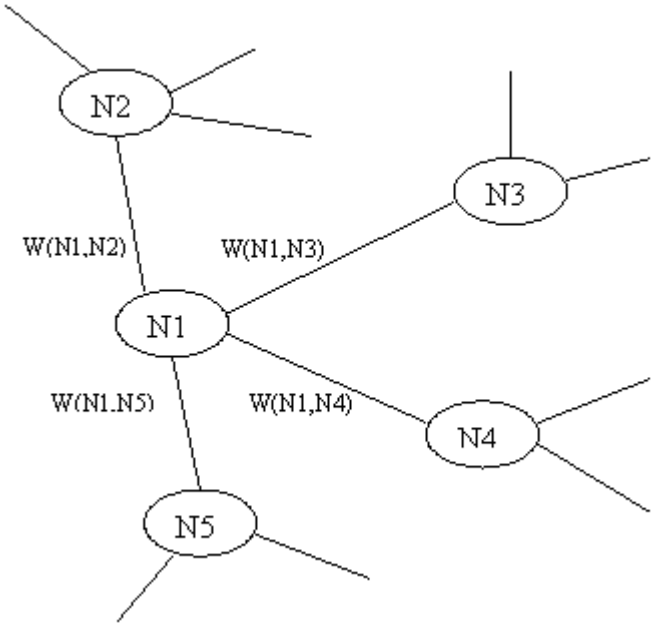

Figure 2: Example of mutual access control in Gnutella

Consider two peers A and B that are connected via Gnutella. Peer B may not necessarily have the files that peer A is interested in. Yet, B does provide service to A by contributing its bandwidth to forward A's queries to the rest of the network. SLIC [6] acknowledges this as a significant contribution. It allows each peer to "rate" its neighbors and to use the ratings to control how many queries from each neighbor to process and to forward on. Peer A would increase the rating of peer B if B provides service to A either directly (by answering its queries) or indirectly (by nodes reachable via B). If A gives a high rating to B, then it would process and forward more queries from B. Conversely, bad service would reduce rating and the number of queries serviced. In other words, SLIC uses this method of rating as a way of retaliation if a node does not contribute or connects to nodes that do not contribute to the P2P network. Hence, this provides the nodes an incentive to share content and also connect to nodes that share content.

To distinguish good neighbors from bad ones, each peer u maintains a weight $W(u, v)$ for each neighbor v where $0 \leq W(u, v) \leq 1$. A weight of 1 indicates an excellent neighbor and a weight of 0 indicates a useless neighbor. A peer decides the level of service that it would provide to each of its neighbors based on the weights associated with them. If a peer u is sharing a total bandwidth b, then

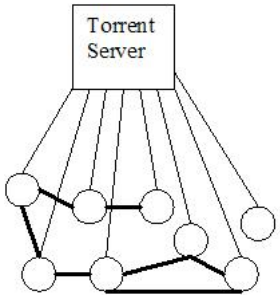

Figure 3: Architecture of Torrents

the piece of bandwidth allotted to neighbor v by peer u, $(B(u,v))$ is given by:

$$B(u,v) = \frac{W(u,v)}{\sum_{x \in N} W(u,x)} * b$$

where N is the set of all neighbors of u. Each peer also updates the weight it associates to all its neighbors periodically. It uses an exponential decay mechanism for updating the weights it associates with each of the neighbors. If $W(u,v)$ denotes the weight u assigns to v in the previous period, $W'(u,v)$ denotes the new weight for the next period and $I(u,v)$ denotes the quality of service from a neighbor v during this period, then:

$$W'(u,v) = \alpha * W(u,v) + (1-\alpha) * I(u,v) \quad where\ 0 < \alpha < 1$$

5 Torrents

5.1 Architecture

The Torrent P2P network [2] is a server based network. All the peers need to be connected to a torrent server (tracker) to be a part of the network. Each tracker tracks only a single file ie it stores information about all the peers that are sharing the file and/or wants to download that file. Each file is split into chunks of 128 KB and each of these chunks are considered as individual resources.

5.1.1 Searching and Resource Location

When a peer wishes to enter the network, it connects to the tracker. So the tracker knows the addresses of all the peers that are currently downloading/sharing chunks of the file that the tracker tracks. Peers are dependent on the tracker to locate the other peers which are sharing chunks of the file. To find the addresses of these peers, it sends a request to the tracker. The tracker returns a random list of the addresses of the peers that are currently sharing chunks of the file.

5.1.2 Seeding

A *seeder* is a peer that has all the chunks of a file. To initialize a tracker for that file, there has to be one or more altruistic *seeders* which upload all chunks of the file to the network. However a *seeder* does not upload all chunks to just one peer but distributes all the chunks among multiple peers in the network. Once the *seeder* uploads all the chunks into the network, it is free to withdraw from the network, although no other peer in the network may have all the chunks of the file. The peers can get all the chunks from each other by giving away/exchanging them among themselves.

5.1.3 Peer to Peer Data transfer

A peer receives the addresses of the other peers which have the partial/full file that it wants to download. It then connects to those peers directly. To each of the peers it connects to, it sends a request for data transfer for the chunks it wants. It also needs to announce the chunks it shares to the peers it connects to. This makes barter/giveaway decisions to be made at the peer level.

5.2 Choking - Default Incentive Mechanism in Torrents

The default incentive mechanism for Torrents is called *Choking* [3]. *Choking* refers to the denial of upload of a shared resource to another peer which sends a request for it. Depending on the bandwidth available to a peer it decides on the number of connections it leaves unchoked. Decisions as to which connections to unchoke are based on the current download rates it receives from the nodes from which it is downloading data. If the download rate from a choked node is higher, then it gets a higher priority to be unchoked. The server does no centralized resource allocation by itself, and hence it is upto the peers to implement it. To avoid bandwidth wastage from repeated choking and unchoking, the decisions are made only after fixed intervals of time. This is called *Restricted Choking*.

As a method of discovering connections which can lead to potentially higher downloads, peers also unchoke a previously disconnected node with a probability p. This makes it possible for a peer to increase its bandwidth utilization to an optimum level. This is referred to as *Optimistic Choking*. It corresponds to the variation of tit for tat for a repeated prisoners dilemma game where the first move is to always cooperate.However, if a node finds that most of the nodes that it is uploading to are choking it, it could snub the nodes which choked it by canceling uploads to those nodes midway.

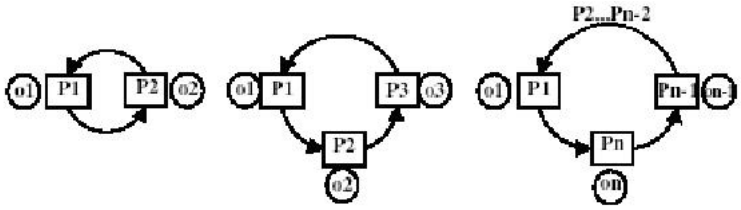

Figure 4: Example of 2,3-way exchange strategies

6 Incentive Mechanisms for Torrents

6.1 Incentive Mechanism with N-way exchanges

A two-way exchange strategy ensures that a resource is provided to by another peer only if that peer provides something in return. In an N-way incentive mechanism [4], a service need not be provided directly to the provider, but more generally priority is given to peers who are part of an N-way cycle exchange of which the provider is a part of.

6.1.1 Increased Network Utilization

If each peer participated only in a two way exchange, and a cylic resource dependency chain existed, only one transfer could take place at one time. This leads to inefficient bandwidth usage as all nodes cannot participate in data transfer at the same time. If a cycle of transfer is established, then all the nodes in the cycle would simultaneously participate in data transfer, leading to higher network utilization.

To implement this incentive mechanism, each node maintains a request tree. A peer with no incoming requests has an empty request tree. An incoming request from another peer also contains the request tree of the requesting peer. The peer then generates a graph of requests from all the nodes that requests resources from it. From this graph it is able to locate resource request cycles involving more than 2 nodes.

Once a cycle has been identified in the generated request tree, the peer passes on a token to the other end of the chain of the identified chain of requests. This notifies the other node of the existence of a complete cycle of requests. If that token is propagated back to the initial peer, passing through all the nodes in the chain, the cycle of data transfer involving the n nodes is initiated.

Example: Consider *Figure 4*. Node A receives the request trees from the nodes $P1$, $P2$ and $P4$ each passing on information about their request trees. From this combined request graph A finds a resource request cycle with three nodes A, $P2$ and $P9$, where $P9$ requests $o6$ which $P2$ shares, $P2$ requests $o2$

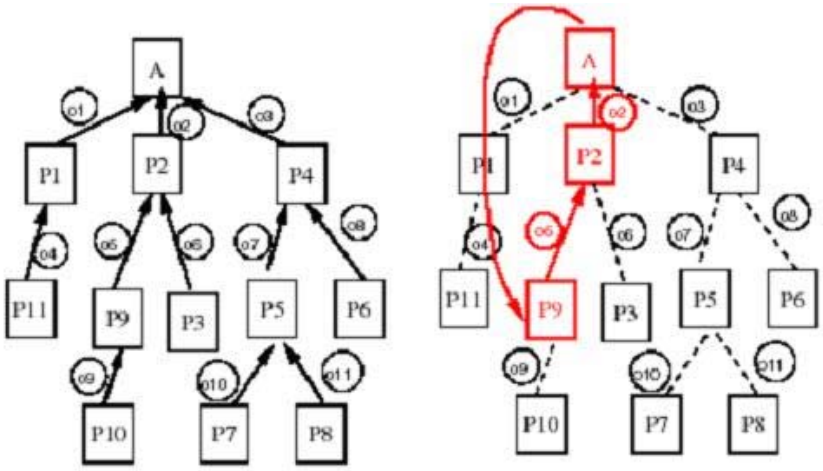

Figure 5: Example of a Resource Request Tree

which A shares and A requests an object which $P9$ shares. A then sends a token to $P9$ indicating the existing of resource request cycle, and this token is passed on by $P9$ to $P2$ which in turn passes it on back to A. Once the token is returned back to A, A starts uploading $o2$ to $P2$, which in turn starts uploading $o6$ to $P9$ which in turn starts uploading to A.

7 Conclusion

In this study we have analyzed the structure of the Gnutella and Torrent Peer to Peer networks. We further analyzed some relevant incentive mechanisms proposed in the literature and suggested methods by which they can be integrated into the existing networks without changing the protocols themselves. Implementation of these incentive mechanisms is expected to curb the problem of *Free-riders* in these networks and hence increase the overall network efficiency. The incentive mechanisms proposed can be implemented at the peer level and do not depend on a centralized server to implement them. This ensures seamless integration of these methods into present networks.

References

[1] Gnutella 0.6 http://rfc-gnutella.sourceforge.netsrcrfc-0_6-draft.html.

[2] Torrent Protocol http://www.bittorrent.com/protocol.html.

[3] Bram Cohen, "Incentives Build Robustness in BitTorrent", http://www.bittorrent.com/bittorrentecon.pdf.

[4] Kostas G. Anagnostakis and Michael B. Greenwald, "Exchange-based Incentive Mechanisms for Peer-to-Peer File Sharing", Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS'04).

[5] Atip Asvanad, Sarvesh Bagla, Munjal H. Kapadia, Ramayya Krishnan, Michael D. Smith and Rahul Telang, "Intelligent Club Management in Peer-to-Peer Networks", Proc. of the Workshop on Economics of Peer to Peer Systems, 2003.

[6] Qixiang sun, Hector Gracia-Molina, "SLIC : A Selfish Link Based Incentive Mechanism for Unstructured Peer-to-Peer Networks", Proceedings of the 24th International Conference on distributed Computing Systems (ICDCS'04).