

Addressing the Non-Cooperation Problem in Competitive P2P Systems

Beverly Yang

Sepandar D. Kamvar

Hector Garcia-Molina

Abstract

The operation of large-scale competitive P2P systems are threatened by the *non-cooperation problem*, where peers do not forward queries to potential competitors. While non-cooperation is not a problem in current P2P free file-sharing systems, it is likely to be a problem in such P2P systems as pay-per-transaction file-sharing systems, P2P auctions, and P2P service discovery systems, where peers are in competition with each other to provide services. Here, we motivate why non-cooperation is likely to be a problem in these types of networks and present an economic protocol to address this problem. This protocol, called the RTR protocol, is based on the buying and selling of the right-to-respond (RTR) to each query in the system.

1 Introduction

While peer-to-peer networks have risen to prominence due to the success of free file-sharing networks like Napster and Kazaa, increasing emphasis is being placed on new applications of P2P, including pay-per-transaction networks, P2P auctions, and P2P service discovery systems. The problems in these networks are likely to be different than the problems encountered in free file-sharing networks.

For example, in the context of free file-sharing, the *freeriding* problem has become a central issue: peers acting in their own best interest conserve their resources (i.e. bandwidth) by sharing no files, and hence, only a small fraction of altruistic users offer almost all the available content. For example, in the Gnutella file-sharing system [2], over 70% of the content was provided by just 5% of the users [1]. As a result, much existing work on incentives in P2P systems have been focused on solving the problem of freeriding (e.g., [3, 4]).

However, in pay-per-transaction systems, freeriding is unlikely to be a problem, since the necessary incentives are inherent in these applications. For example, in a pay-per-transaction file-sharing system where peers get paid for uploading files, peers will want to share files, because this generates income. In

an auction system, where the auction advertisement is analogous to a query and bids analogous to query responses, peers will want to submit bids. Even in a free file-sharing system where users share their original music or artwork, users have an incentive to share the files in order to increase their publicity.

In each of the systems described above, not only are peers eager to provide services (e.g. share files), but they are in *competition* with other peers to provide their services. Competition is a problem in P2P frameworks that rely on peers to forward queries (e.g., Gnutella [2], DHTs like [6], landmark routing [8], etc),¹ because a peer acting in its own best interests will not forward queries to potential competitors. For example, a peer providing a car rental service might not forward a query for car rental services. Instead, it could answer the query and then drop it, so as to improve its chances of gaining business. Therefore, P2P systems will no longer operate correctly due to non-cooperation, even though abundant services are available.

In this paper, we propose an economic protocol to ensure that peers cooperate in the *operation* of P2P systems in the face of competition. We present this protocol in Section 2, where we discuss how our proposed system will not only assure proper operation, but can also improve the efficiency and effectiveness of the search mechanism as well. In Section 3, we present a high-level attack analysis of the protocol, including a discussion of potential attacks and pitfalls, and how these issues are countered. Finally, in Section 4 we present a list of important areas requiring further investigation.

In the following discussion, we will illustrate our protocol on top of the Gnutella protocol for P2P search running a pay-per-transaction file-sharing application. Important items for future work include extending these ideas to architectures other

¹The exceptions are systems that require no forwarding, such as Napster (<http://www.napster.com>).

than Gnutella, such as DHTs (see Section 4). Also, the following discussion assumes the existence of an efficient micropayment scheme for P2P systems, such as that described in [7].

1.1 Preliminaries

The basic Gnutella search protocol works as follows: each user runs a client (or *peer*), which is connected to a small number of other peers (known as *neighbors*) in an overlay network. When a user submits a query, her peer will send the query message to all its neighbors, who will in turn forward the query to their neighbors, and so on. A peer that receives a query and finds that it can answer will send a response to the querying peer.² The querying peer will wait a period of time for responses to arrive, and then it will select one or more responding peers from which to buy services. In a pay-per-transaction application, the service offered is the download of a file, and the querying peer will pay the selected peer(s) for each download transaction. The price per download may vary depending on the file. Further details of the Gnutella protocol can be found in [2]. Clearly, if peers do not forward queries, the search mechanism will fail.

2 RTR Protocol

At the core of our protocol is the concept of a *right to respond*, or RTR. An RTR is simply a token signifying that a peer has a right to respond to a query message. We choose this name (“right to respond”) in order to emphasize that a query is really a commodity. Peers should pay to receive the query, because that in turn brings in potential business. If a peer never receives any queries, then it can never provide its service to anyone. An analogous concept in real-life markets are companies who buy lists of emails or referrals from other companies, so that they have a new pool of potential customers.

Once a peer buys an RTR for a given query, it may do one or both of the following: (a) respond to the query and hope that it is chosen to upload its services,

²In Gnutella, response messages are actually forwarded along the reverse path traveled by the query. In systems that do not require anonymity, sending responses directly to the querying peer is more efficient.

and (b) sell the RTR to other peers.³ Peers can buy and sell RTRs with their neighbors only.

In this framework, selling an RTR is equivalent to forwarding a query. Hence, there is built-in incentive to forward queries, since peers get paid to do so. Of course, some peers may still choose to not forward any queries in order to increase the probability that they will be chosen to provide the service. However, their actions will be offset by those peers who hedge their risk by selling a few RTRs, and by those peers who speculate in RTRs (buying RTRs simply to resell them).

An RTR has the following format:

$$RTR = \{Q, ts, query\}_{SK_Q} \quad (1)$$

where Q is the identity of the querying peer, ts is the timestamp at which the query was first issued, and $query$ is the actual query string. These three values are signed by the querying peer’s secret key SK_Q , so that RTRs cannot be forged. Hence, each query requires a single signature generation, and one verification per forward.

2.1 Subscriptions

Because RTRs are inexpensive, it is crucial to keep the overhead of buying and selling at a minimum. If neighbors were to negotiate the price of each RTR transferred, the overhead of the transaction would exceed the value of the RTR. In order to minimize transaction overhead, we introduce the idea of RTR *subscriptions*. Rather than have a peer pay its neighbor for each individual RTR, it “bulk-orders” a subscription of RTRs over time. A subscription is described by three details:

- The *cost* (e.g., \$/day) of the subscription.
- The *rate* (e.g., messages/day) of RTRs that will be delivered.
- *Filters* that specify the “content” and “quality” of the RTRs that will be sold in the subscription.

An apt analogy to RTR subscriptions is magazine subscriptions, where the rate, cost, and content (e.g., articles) fully describe the subscription.

Filters can be set on any of the three fields of an RTR: the reputation of the querying peer Q , the timestamp, or the query string. Filters on the query string specify the *content* of the RTR, and affect the

³If a peer sells an RTR, it may still respond to the query corresponding to the RTR. That is, a peer does not lose the right to respond to a query when it sells the RTR for that query.

probability that the subscriber can respond to the query. Content filters may have varying levels of restrictiveness. For example, a content filter may specify that the RTRs sold in a particular subscription will all be queries for a particular genre of music files. Or, a content filter may specify exactly what RTRs (i.e., for which exact files) will be sold in a particular subscription.⁴ Filters on the reputation of querying peer Q and the age of the query (indicated by the timestamp) specify the *quality* of the RTR, and affect the probability that a responding peer will be chosen and paid for its services.

In establishing subscriptions, a peer A will have an idea of what RTRs it wants to receive, and at what rate (cost will be described in Section 2.2). For example, A may have a large collection of classical music, and is therefore interested in many queries (high rate) for that genre of music. Or, A may only want to receive queries to which it can respond (more restrictive content filter). When A negotiates with a potential neighbor B , A communicates its preferences to B . Peer B may then state the cost of the subscription. If the price is unacceptable or if B cannot meet A 's preferences, then no subscription is established. Note that, just as in real life one may need several different magazine subscriptions to satisfy one's needs, a peer may need to subscribe from several neighbors so that the combined subscriptions meet its preferences.

Continuing with the magazine analogy, not all articles in an issue will be relevant to the reader. If too many articles are irrelevant or of poor quality, then the reader may not renew his subscription. Likewise, if a peer receives too many RTRs that violate content filters or are "borderline" in quality (e.g., as old as they can possibly be without violating the constraint), then that peer can cancel its subscription by disconnecting from its neighbor. It is therefore in each seller's interest to provide high-quality, relevant RTRs to its best ability – otherwise, it may lose business (see Section 2.3). Furthermore, successful magazines need to develop a good reputation. Likewise, along with reputations for being a good consumer and producer of services, peers build up reputations as vendor of good RTRs. Such reputations help peers choose good neighbors.

⁴Indeed, such a filter results in a *super-peer* relationship in which a query is only forwarded from super-peer to client if the client can definitely answer the query.

Note that since brand name and reputation are so important to the success of a peer, common issues frequently associated with P2P systems – zero-cost identities and the need for anonymity – are much less of an issue in the applications we consider. In order to build reputation and brand name, users are likely maintain a small number of public identities, and use those identities over time. Furthermore, if "real money" is used for payment of services, identities will be linked to real-world constructs such as credit cards, making it impractical or impossible to create many identities.

2.2 Pricing

RTR vendors may price subscriptions as they see fit, but should keep in mind the following considerations in determining the offer price of subscriptions.

First, several factors contribute to the value of an RTR to an individual peer A .

- *Peer A 's ability to respond to the query for which the RTR is purchased.* If peer A does not own a file to match the query, it is not able to gain income by responding to the RTR.
- *Whether peer A is chosen to upload the file.* The querying peer is likely to choose only one responder from which to download the file it seeks. Therefore, if many peers buy an RTR for a particular query and respond, then the probability that peer A is chosen decreases.
- *The price of the file for which the RTR is purchased.* Peer A will gain more income by responding to a query for an expensive file than for a cheap file.
- *Peer A 's ability to resell an RTR in a subscription.* Peer A may resell if the RTR fits its subscribing neighbors' content and quality filters.

Peer A will generally be willing to pay more for a subscription that it expects to contain high-value RTRs. Therefore, a RTR vendor may increase the price of an RTR subscription that has increased restrictiveness on its filters, since content filters increase the probability that the subscribing peer owns the file that is being queried for, and quality filters increase the probability that a peer will be chosen for upload. Such a pricing scheme reflects intuition, since neighbors will need to expend greater effort in sorting through RTRs that match the filters, and fewer RTRs will qualify. By paying a premium for filters, peer A is paying for the increased labor and

risk on the part of the vendor and the decreased risk on the part of peer *A*.

Another characteristic that may affect the price of an RTR subscription is the reputation of the vendor. Peer *A* is likely willing to pay more for a subscription from a vendor with whom it has had previous satisfactory experience (or with whom a friend has had satisfactory experience). Again, peer *A* is paying for decreased risk on its part and the labor on the part of the vendor that went into building its reputation.

2.3 Impact of Quality and Content Filters

Not only can filters help individual peers by reducing the risk in buying RTRs, but they can also have globally beneficial effects as well. Recall that peers can cancel bad subscriptions by disconnecting from their neighbors; hence, it is in a peers interest to forward only high quality, relevant RTRs to its best ability – otherwise, it loses business. Here, we discuss how allowing peers to choose neighbors based on quality and content of RTRs results in good system performance.

(1) **Age.** Newer queries are preferred over older queries because they increase the chance that one can reply before the querying peer “closes” the query (e.g., closing the auction, selecting someone to download from, etc). Furthermore, a lower age decreases the chance that the RTR is a duplicate (i.e., received earlier from a different neighbor). Hence, peers should not forward an RTR with an age beyond a given threshold (otherwise, again, its neighbor will drop the connection due to the low quality RTRs). Like the *time-to-live* construct in Gnutella, the effect of dropping old RTRs will limit the size of the network that receives the query. This effect is important in order to prevent inefficient flooding of the network.

(2) **Reputation.** The goal of reputation systems is to allow users to avoid dealing with other users that are malicious (e.g., return corrupt files) or provide poor service (e.g., often die before a transaction is completed). In our scheme, by not forwarding RTRs belonging to low-reputation querying peers, we are ensuring that those who provide bad service are penalized with fewer results for their own queries. Global reputations in P2P systems can be managed, for example, by a system like EigenTrust [5]. Note that a reputation system is not necessary for the RTR protocol. In the absence of a reputation system, the quality of RTRs may be judged by age and relevance

only.

(3) **Content.** Peers will connect to other peers who can provide them the RTRs that they want. Thus, content filters serve as “routing hints” by which queries can be routed to those peers containing relevant data, thereby resulting in more *efficient* search. By buying RTR subscriptions with appropriate content filters, peers will connect to those peers who can provide them queries to which they are able to respond. Furthermore, perhaps a peer *A* is unhappy with the subscription that its neighbor provides because, for example, the subscription has too much hard rock and too little punk rock. In this case, *A* can disconnect and eventually find another neighbor whose sells subscriptions that has more punk. Hence, we hope that communities, or “clusters” in the topology, will form around interests, which further lends itself to efficient, effective search.

In summary, by simply acting out of self-interest, peers in our scheme can form a forwarding policy that enforces reputation, limits flooding, and promotes intelligent routing/topology formation.

3 Attack Analysis

Bogus Queries. The main attack possible on the proposed incentive system is that of greedy peers generating bogus queries for services they do not really need. The motivation for doing so is to earn money by selling the initial RTRs to its neighbors.

The primary counter to this type of attack are the notions of reputation and past experience. That is, if a peer is unsatisfied with the RTRs that are being included in a subscription, they may disconnect from that peer and buy a subscription from a new peer. Furthermore, peers are unlikely to buy expensive subscriptions from vendors with whom they have had no experience. Finally, if the peer gains a good reputation by consistently providing RTRs that lead to uploads, it will be able to charge a premium for its subscription packages. Therefore, there is significant incentive not to sell RTRs for bogus queries to one’s neighbors, especially in the presence of a global reputation system.

Another way to counter this attack is to make users pay to submit new queries. If there exists a lightweight centralized broker (which is probably required for any system using real money for payment), then a peer can buy a “right to query” (RTQ) token from the broker. An analogy to this idea is

paying the owner of a website or billboard to post an advertisement. Each time that peer sends out a query, it must include a new RTQ along with its RTR messages. If the peer reuses a token, then anyone who receives two distinct RTRs (queries) with the same RTQ (right to query), can report the RTRs to the broker as evidence of foul play.

If no centralized broker is available, we can still limit queries via a distributed quota enforcement system (e.g. [7]). Each peer is allotted a “quota” of RTQs (e.g., one query per day). RTQs can accumulate over time if not used. A peer can choose to use the RTQ (regardless of whether it really needs a service), or else sell it to peers who wish to buy it.

Note also that malicious users cannot send out a bogus RTR that lists a different peer as the querying peer, because each RTR is signed by the querying peer.

Blasting. Another potential pitfall of our scheme is that a peer may connect to many, many neighbors, and then forward RTRs to all of them whenever it receives an single RTR from one of them. In this way, the peer, or “blaster”, can greatly multiply its money.

However, recall that if a peer is unsatisfied with the RTRs it is receiving from a neighbor, it can simply drop its neighbor. Hence, if the blaster is indiscriminately forwarding low quality RTRs to all its neighbors, it will eventually lose all its customers. Because it has gained a bad reputation, it will also be unable to find new customers. Furthermore, it is difficult for any peer to gain many neighbors in the first place, unless that peer first establishes a reputation for selling good RTRs. Any peer that has established such a reputation will benefit in the long run by continuing to provide good RTRs, rather than becoming a blaster and destroying its reputation.

4 Future Work

There are many areas in the RTR protocol that require further study. First, to properly evaluate our incentives system, we need utility functions to describe what users value. The peers in our system must weigh a number of considerations, such as what types of RTRs it wants to sell, how it wants to price subscriptions, whether it wants to make money by uploading or selling RTRs, etc. Defining a behavior model that is descriptive enough to capture all

the different preferences, while remaining practical enough for efficient simulation, is a challenge.

Second, in our current model, the willingness of peers to pay for RTRs (because they represent potential business) “pull” queries through the network. We may also want to consider a “push” model in which the querying peer can pay peers to forward its query, with a guarantee that a minimum number of peers will eventually receive the query.

Finally, we would also like to address the non-cooperation problem over different search mechanisms, such as DHTs. DHTs pose new challenges because peers can no longer choose their neighbors, nor do peers have a choice as to which queries they forward. Due to the rigid, deterministic topology and routing policy of DHTs, the “push” model of forwarding queries may be more appropriate in ensuring that messages are properly routed.

In summary, we believe that the *non-cooperation problem* will present a significant challenge to competitive P2P systems. We present one possible protocol based on the notion of RTRs (right to respond) that gives peers the incentive to cooperate in the operation of the system, even in the face of competition for providing services. In addition, our scheme has the potential to improve the efficiency and effectiveness of the search mechanism, even though all peers act out of self-interest.

References

- [1] E. Adar and B. Huberman. Free Riding on Gnutella. http://www.firstmonday.dk/issues/issue5_10/-adar/index.html, 2000.
- [2] Gnutella website. <http://gnutella.wego.com>.
- [3] P. Golle, K. Leyton-Brown, I. Mironov, and M. Lillibridge. Incentives for sharing in peer-to-peer networks. In *Proc. ACM Conference on Electronic Commerce*, 2001.
- [4] S Kamvar, M. Schlosser, and H. Garcia-Molina. Incentives for Combatting Freeriding on P2P Networks. Technical report, Stanford University, 2003.
- [5] S. Kamvar, M. Schlosser, and H. Garcia-Molina. The EigenTrust Algorithm for Reputation Management in P2P Networks. In *In Proc. WWW*, 2003.
- [6] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proc. ACM SIGCOMM*, 2001.
- [7] B. Yang, S. Kamvar, and H. Garcia-Molina. Secure Score Management in P2P Systems. Technical report, Stanford University, 2003.
- [8] B. Zhao, Y. Duan, L. Huang, A. Joseph, and J. Kubiatowicz. Brocade: Landmark routing on overlay networks. In *Proc. IPTPS*, 2002.