A Distributed and Measurement-based Framework Against Free Riding in Peer-to-Peer Networks

Murat Karakaya, Ibrahim Korpeoglu, Özgür Ulusoy
Bilkent University, Dept. of Computer Engineering
06800 Ankara, Turkey
{muratk, korpe, oulusoy}@cs.bilkent.edu.tr

Abstract

In this paper,[1] *we propose a distributed and measurement-based method to reduce the degree of free riding in P2P networks. We primarily focus on developing schemes to locate free riders and on determining policies that can be used to take actions against them. Our proposed schemes require each peer to monitor its neighboring peers, make decisions if they exhibit any kind of free riding, and take appropriate actions if required.*

1 Introduction

As a P2P computing concept, *free riding* means exploiting P2P network resources (through searching, downloading objects, or using services) without contributing to the P2P network at desirable levels. Researchers have observed the existence of high degrees of free riding in P2P networks and they suggest that free riding may be an important threat against the existence and efficient operation of P2P networks [1].

In this paper, we propose two schemes to cope with free riding in unstructured P2P networks such as Gnutella [2]. The first scheme primarily focuses on locating and detecting free riders, whereas the second one deals with taking actions against them.

2 Locating Free Riders

We propose a system in which every peer passively monitors the activities of its neighboring peers. In the proposed system, peers can be classified into two different roles. In the first type of role, a peer functions as a monitoring peer, P_M, which monitors and records the number of messages coming from and going to neighboring peers. At the same time, each peer is a controlled peer, P_C, which means that its messages are monitored and counted by its neighboring peers.

The information that is maintained about neighbors of a peer (P_M) consists of some statistical counters which are presented in Table 1. These counters are updated when messages are received from the neighbors and when messages are sent towards the neighbors.

Table 1. Observed Counters

Symbol	Description
Q_P	Number of `Query` descriptors submitted by P_C.
RQ_P	Number of `Query` descriptors routed by P_C.
TQ_P	Number of `Query` descriptors routed towards P_C.
QH_P	Number of `QueryHit` descriptors submitted by P_C.
RQH_P	Number of `QueryHit` descriptors routed by P_C.
SQH_P	Number of `QueryHit` descriptors satisfying queries of P_C.

Below, we identify some possible free riding types that a peer may exhibit. We also formulize how the identified free riding types can be detected by using the statistical information (shown in Table 1) gathered about a free riding peer.

- **Type 1 - Non-contributor: Peer does not share anything at all or shares uninteresting files.** It may be observed that a neighboring peer does not return any `QueryHit` messages to the queries that it receives. Whenever the monitoring peer initiates a search or routes a search on behalf of other peers by sending a `Query` message to its neighbors, the monitoring peer also increases the value of the respective TQ counters (maintained in a log table) for its neighbors. The monitoring peer also observes and counts the `QueryHit` messages received from the neighboring peers. If the monitoring peer receives a `QueryHit` message that has the IP address of one of its neighbors in it, the monitoring peer increases the value of the QH counter maintained for that peer in the log table.

  The monitoring peer then compares the values of TQ and QH counters maintained for a neighboring

[1] This work is supported in part by The Scientific and Research Council of Turkey (TUBITAK), Grant EEEAG-103E014.

peer to decide if that peer is a free rider that is not sharing any files (a *non-contributor*). More specifically, for this decision to be made, the monitoring peer may compare the QH/TQ ratio against a threshold value and decide that the neighbor is a free rider of type non-contributor if the ratio is smaller than the threshold. Below, we formulize the condition that is required to judge if a neighboring peer is a free rider or not.

Furthermore, to remove the warm-up period and to obtain valid statistical information we propose to use a threshold value, τ_{TQ}, for the number of forwarded `Query` messages (TQ_P) to the peer that is observed. Only if that threshold is exceeded, the counter values are used to infer free riding.

```
if ($\mathrm{TQ}_P > \tau_{TQ}$) and ($\frac{QH_P}{TQ_P} < \tau_{non\_contributor}$) then
  peer P is considered as a non-contributor
endif
```

- **Type 2 - Consumer: A peer consumes more resources than that it shares.** A monitoring peer counts the `QueryHit` responses (QH) originated from its neighbors and successful `QueryHit` messages (SQH) destined to and received by its neighbors. The comparison of these two numbers reveals if any of the neighboring peers consumes more than it shares. More specifically, a threshold value, $\tau_{consumer}$, can be compared against the ratio of these two numbers to decide if the neigboring peer is a free rider of type consumer or not.

```
if ($\mathrm{TQ}_P > \tau_{TQ}$) and ($\frac{QH_P}{SQH_P} < \tau_{consumer}$) then
  peer P is considered as a consumer
endif
```

- **Type 3 - Dropper: A peer drops others' queries.** A monitoring peer counts `Query` and `QueryHit` messages forwarded by each of its neighbors. If these two values (RQ and RQH) are very low for a neighboring peer, it can be assumed that the neighboring peer does not have enough connections or it drops queries and/or query hits. We call this type of free rider as a *dropper*. The ratio of sum of RQ and RQH counters to the value of TQ counter is compared against a treshold value, denoted with $\tau_{dropper}$, to decide if a neighboring peer is a dropper or not.

```
if ($\mathrm{TQ}_P > \tau_{TQ}$) and ($\frac{RQ_P+RQH_P}{TQ_P} < \tau_{dropper}$) then
  peer P is considered as a dropper
endif
```

3 Actions against Free Riders

If a peer identifies another peer as a free rider, it can take some counter-actions against it. We specify three levels of counter-actions. Level 1 action is the least restrictive one for the free rider, whereas level 3 action is the most restrictive one.

- **Level 1 Action: Decrementing TTL value more than one.** To act against a suspected free rider, the monitoring peer can play with the TTL value for `Query` messages that are received from the suspected peer, i.e. it can decrement the TTL value by more than one before forwarding. In this way, the search horizon of the free riding peer is narrowed down. This also reduces the overhead that `Query` messages may impose on the network. This counter-action is applied to a peer that exhibits only one type of free riding, i.e. the peer is either a non-contributor, or a dropper, or a consumer.
- **Level 2 Action: Ignoring requests.** A free rider peer can be punished by the monitoring peer by ignoring the searches (i.e. the `Query` messages) originating from that free riding peer. This counter-action is applied to a peer that is exactly exhibiting two types of free riding (for example, to a peer that is both a consumer and a dropper).
- **Level 3 Action: Disconnecting from network.** If a peer is sure that a neighboring peer is a free rider that is exhibiting all types of free riding, the peer may drop the connection with that peer. When disconnection is executed, the disconnected peer should reconnect to the system through a new peer if it wants to benefit from the network as a legitimate peer.

In [3], we provide a finite state machine (FSM) abtraction that shows the current type of behaviour of a neigboring peer as states (hence in each state a different counter-action can be applied). The FSM model also provides the transitions between states and the conditions that trigger the transitions.

4 Conclusion

In this paper we have proposed a distributed and measurement based method to reduce the degree of free riding in unstructured P2P networks. We are currently developing a simulation program to implement and evaluate the proposed schemes.

References

[1] E. Adar and B. A. Huberman. Free riding on gnutella. *http://www.firstmonday.dk/issues/issue5_10/adar/*, 2000.

[2] Clip2. The gnutella protocol specification v0.4 (document revision 1.2). *http://www9.limewire.com/developer/gnutella protocol0.4.pdf*, Jun. 2001.

[3] M. Karakaya, I. Korpeoglu, and Ö. Ulusoy. A distributed and measurement-based framework against free riding in peer-to-peer networks. *Technical Report BU-CE-0405, Department of Computer Engineering, Bilkent University*, May 2004.