

# Collaboration-Aware Peer-to-Peer Media Streaming

Song Ye

yesong@cs.dartmouth.edu

Fillia Makedon

makedon@cs.dartmouth.edu

Dartmouth Experimental Visualization Laboratory  
Department of Computer Science, Dartmouth College  
Hanover, NH 03755

## ABSTRACT

Peer-to-Peer (P2P) media streaming has emerged as a promising solution to media streaming in large distributed systems such as the Internet. Several P2P media streaming solutions have been proposed by researchers, however they all implicitly assume peers are collaborative, thus they suffer from the selfish peers that are not willing to collaborate. In this paper we introduce an incentive mechanism to urge selfish peers to behave collaboratively. It combines the traditional reputation-based approach and an online streaming behavior monitoring scheme. Our preliminary results show that the overall performance achieved by collaborative peers do not suffer from the existence of non-collaborative peers. The incentive mechanism is orthogonal to the existing media streaming solutions and can be integrated into them.

## Categories and Subject Descriptors

C.2.4 [Computer-Communication Networks]: Distributed Systems—Distributed Applications; H.3.5 [Information Storage and Retrieve]: Online Information Services—Data Sharing

## General Terms

Design, Performance

## Keywords

Peer-to-Peer, Media Streaming, Collaboration, Reputation

## 1. INTRODUCTION

Advances in computers, networking and communications have created new distribution channels for the dissemination of multimedia content. Recently P2P media streaming systems have attracted tremendous attention from both industry and academia. In a P2P media streaming system, a subset of peers own a certain media file, and they stream the media file to requesting peers. On the other hand, the

requesting peers playback and cache the media data during the streaming session, and they can stream the cached media data to other peers if requested. Multicast is a natural solution to media streaming systems.

Application-layer multicast [6] has been widely accepted by media streaming systems recently. It provides more capabilities than IP multicast, since every node in application layer is able to cache and forward media streams, while a router in IP multicast can only forward media packets. Several application-layer multicast based P2P media streaming systems have been proposed [13, 10, 12, 7]. However, they focus on how to efficiently build and maintain application-layer multicast structures (trees, meshes, or overlays); none of them have ever discussed the possibility of keeping the media streaming performance with the existence of non-collaborative peers (also known as *free-riders* [1]), which largely exist in real P2P systems, including both file sharing and media streaming systems. There are some research works regarding how to identify and prevent non-collaborative peers in P2P file sharing systems; however, due to the inherent differences between P2P file sharing and media streaming, they can not be applied directly to P2P media streaming systems. We are aware of a similar work [4], which proposes a rank-based peer selection mechanism. It is more like a reputation-based approach and focuses on peer selection, while our approach focuses on online streaming behavior monitoring and tries to maintain a satisfiable level of media streaming for all collaborative peers.

## 2. SYSTEM MODEL

We assume a P2P on-demand media streaming system with  $N$  peers:  $P_1, P_2, \dots, P_N$ . Peers can be uniquely identified by their IDs and public-private key pairs. A peer  $P_i$  has its inbound bandwidth  $B_i^{in}$  and outbound bandwidth  $B_i^{out}$ . In this paper, we do not consider the network topology<sup>1</sup>, and therefore the construction of media multicasting tree is determined by the inbound and outbound bandwidth of peers. We assume there is only one media clip  $C$  to be streamed in the system. The set of peers that can act as media servers for  $C$  is defined as  $S(C) = \{P_{j_1}, P_{j_2}, \dots, P_{j_k}\}$  and  $|S(C)| \geq 1$ . During a media streaming session,  $S(C)$  is dynamically changed as a peer with large storage space may cache  $C$  and act as a server peer for  $C$ . We do not

<sup>1</sup>Network topology is actually a key issue in constructing a media streaming overlay. Although some previous P2P media streaming research did not consider this issue, recently topology aware P2P media streaming approaches have been investigated [5, 14].

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to publish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Copyright 200X ACM X-XXXXX-XX-X/XX/XX ...\$5.00.

differentiate media servers from other peers; a server peer  $P_i$  for  $C$  simply does not request media streams from other peers.

We use  $P_i \rightarrow P_j$  to denote that  $P_i$  is sending a media stream to  $P_j$ .  $P_i$  is called the *upstream peer* of  $P_j$  and  $P_j$  is the *downstream peer* of  $P_i$ .  $D(P_i) = \{P_j | P_i \rightarrow P_j, 1 \leq j \leq N\}$  denotes the set of all the peers to which  $P_i$  sends media streams;  $U(P_i) = \{P_j | P_j \rightarrow P_i, 1 \leq j \leq N\}$  denotes the set of all the peers from which  $P_i$  receives media streams.

We use Multiple Description Coding (MDC) to encode the media content being streamed. MDC encodes the media content into  $M > 1$  separate streams (or *descriptions*), such that any  $m < M$  descriptions can be used together to be decoded into a media signal with some distortion, which is decided by  $m$ ; the more descriptions received, the higher the quality of the reconstructed media.

### 3. COLLABORATIVE MEDIA STREAMING

In this section, first we discuss collaboration in a P2P media streaming system, and then we introduce a mechanism to promote collaboration among media streaming peers. Finally we will discuss the performance of the proposed online monitoring mechanism and give some preliminary experimental results.

#### 3.1 Collaborations in P2P Media Streaming Systems

P2P systems rely on collaborations among self-interested peers. For example, in a file-sharing system, overall download latency and failure rate increase when peers do not upload their files. In a wireless ad-hoc network, overall packet latency and loss rate increase when nodes refuse to route packets on behalf of others. Further examples are digital collection preservation, overlay routing, *etc.* P2P media streaming is no exception. If most peers only want to receive media streams from others without contributing its local storage space and outbound bandwidth to cache and forward streams, the whole streaming session can not survive.

In a P2P media streaming system, collaborative peers are willing to follow the media streaming protocol and contribute their local resources to cache and forward media streams. However, there exist non-collaborative peers, which are also called *selfish peers*. The goal of selfish peers is to maximize their own benefits. Specifically, a selfish peer participating a media streaming session tries its best to receive media streams from upstream peers, as well as to avoid caching and streaming media to other peers to save its local resources. For example, a peer may intentionally exaggerate its outbound bandwidth to make other peers to prefer to send it media streams (as it has the ability to forward media streams to other peers), but instead it does not send media streams to other peers at all or only uses a very limited part of its outbound bandwidth for forwarding. Another example is media streaming among mobile devices. A mobile device with limited storage space and battery capacity may attempt to avoid caching and forwarding media streams.

To assure the quality of the streaming service, when a peer is selecting its downstream peers, it should give preference to the peers which are identified to be collaborative. Existing trust and reputation management approaches [9, 2], which have been investigated in a P2P file sharing environment, can be applied to help select *trusted/collaborative*

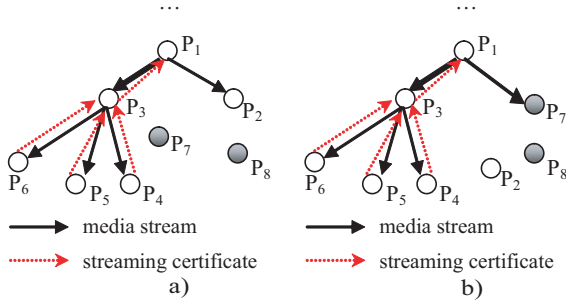
peers with good *reputation*. In those reputation systems, when every peer interacts with others, it records the experience and develop a reputation rating for each other. For example, in a P2P file sharing system, a peer keeps record of total number of files it has downloaded from another peer and how many of them are authentic. Peers share their reputation opinions with other peers; generally,  $P_i$ 's trust evaluation about  $P_j$  is determined by  $P_i$ 's own experience (if  $P_i$  has interacted with  $P_j$ ) and other peers' opinion about  $P_j$ .

However, P2P file sharing and P2P media streaming are inherently different, although both of them are some kind of content sharing systems. The major difference between P2P file sharing system and a P2P media streaming system lies in the data sharing mode among peers: the former uses the *open-after-downloading* mode, while the latter uses the *play-while-downloading* mode. Compared with a P2P file sharing session, a media streaming session always has a time constraint. For example, if a media clip  $C$  is streamed in a P2P network, once  $P_i$  starts the playback of  $C$ , it wants to maintain a certain level of streaming speed, otherwise it may declare it fails in the media streaming session. However, if  $C$  is shared in a file sharing system,  $P_i$  requesting  $C$  does not care about a sustained download speed; it only cares about the total downloading time and whether the downloaded file is corrupted or not. A media streaming peer is able to verify the content and playback quality of a media stream on-the-fly, while a file sharing peer can verify the content of a file only after the peer has downloaded it. Therefore in a P2P media sharing system, we not only need to management reputations among the peers, we also need to monitor the behavior of media streaming peers in a timely manner.

It should be noted that there also exist malicious peers that are different from selfish ones. They are controlled by an outside adversary, and their goal is to compromise the whole system. They may intentionally corrupt the media stream by dropping chosen packets, rearranging the order of the packets in an arbitrary way, and injecting new packets into the transmitted stream. Some recent approaches such as [8] have addressed this kind of peers. Although those approaches are not designed specifically for media streaming systems, they assure the integrity of the content being transferred in the system and will work for media streaming systems. Malicious peers usually collude with each other. The best way to deal with malicious peers is to identify them and keep them out of the media streaming system. The incentive mechanism introduced in this paper can also be applied to alleviate the problem of malicious peers, but we will not discuss the details here.

#### 3.2 An Incentive Mechanism for Collaboration

To maximize the efficiency of a media streaming session, every collaborative peer is willing to transfer media streams to the requesting peers, especially to those peers that can cache and forward streams to other peers. However, to save their outbound bandwidth, local storage space, and processing power, selfish peers only want to receive streams without forwarding. We need some incentive mechanism to urge selfish peers to contribute their local resources, and if they do not contribute, they should suffer from a poor quality of streaming services. Due to the inherent realtime requirements of media streaming applications, the incentive mech-



**Figure 1:** a)  $P_2$  is a selfish peer; b)  $P_1$  discards  $P_2$  and selects  $P_7$

anism should be effective in a timely manner.

We assume there is a reputation system for the P2P media streaming network. Every peer’s trustworthiness (whether it is collaborative or not) can be evaluated and propagated distributively, as in a P2P file sharing system. The principle behind our incentive mechanism is simple. During a media sharing session, a peer  $P_i$ ’s behavior is monitored by both its upstream peers  $U(P_i)$  and downstream peers  $D(P_i)$ . If  $P_i$  is identified to be selfish, punishment is applied by  $U(P_i)$ . A peer  $P_j$  can dynamically select its downstream peers  $D(P_j)$  and adjust its streaming speed to  $D(P_j)$  based on its evaluation about the peers in  $D(P_j)$ .

Figure 1.a shows a partial multicast tree with  $P_1$  as the “root” peer.  $P_1$  is sending media streams to both  $P_2$  and  $P_3$ .  $P_2$  is a selfish peer and is not willing to forward streams it received to other requesting peers  $P_7$  and  $P_8$ , whereas  $P_3$  is collaborative and it forwards the media content to three peers  $P_4$ ,  $P_5$ , and  $P_6$ . Since  $P_4$ ,  $P_5$ , and  $P_6$  are receiving streams from  $P_3$ , they will issue *streaming certificates*  $\mathbf{SCert}(P_i, P_3)$ ,  $i = 4, 5, 6$  periodically to  $P_3$  stating the fact that they have received streams from  $P_3$ , which in turn shows these certificates to  $P_1$ .

In every time interval  $\tau$ , a streaming certificate  $\mathbf{SCert}(P_i, P_j)$  should be issued by  $P_i$  if  $P_i \in D(P_j)$ . The certificate has the following form:

$$\mathbf{SCert}(P_i, P_j) = \langle P_j, t_s, t_e, S \rangle_{P_i}$$

It is signed by  $P_i$ ’s private key and certifies that from time  $t_s$  to  $t_e$ ,  $P_j$  correctly send the stream of size  $S$  to  $P_i$  as requested. Usually we have  $t_e - t_s = \tau$ , however, when a downstream peer leaves before the end of an interval (it may have received what it requires), it can issue the streaming certificate that has  $t_e - t_s < \tau$ . Not all streaming certificates are equally important; a certificate issued by a peer with high reputation is of more importance than the one issued by a selfish peer.

In Figure 1,  $P_1$  can tell whether  $P_2$  and  $P_3$  have forwarded media streams to other peers by periodically asking them to show the streaming certificates they have collected from their downstream peers. Since  $P_2$  has no downstream peers, it is identified by  $P_1$  as a selfish peer very quickly.  $P_1$  can reduce the streaming speed to  $P_2$  (increase streaming speed to  $P_3$  provided  $B_3^m$  is not saturated) or simply refuse to supply  $P_2$  streams as  $P_7$  and  $P_8$  are still requesting for streams. As shown in Figure 1.b,  $P_1$  will no longer send streams to  $P_3$ , in-

stead it selects  $P_7$ . Similarly,  $P_3$  also issue  $\mathbf{SCert}(P_3, P_1)$  to  $P_1$ , which in turn shows the certificate to its own upstream peers (not shown in the figures).

### 3.3 Discussion

The above approach creates strong incentives to urge selfish peers to collaborate, because if they want to receive media streams from upstream peers, they have to forward streams to other requesting peers to earn streaming certificates. A peer’s selfish behavior can be detected by its upstream peers in a timely manner. If a peer has been identified as selfish, it is not simply excluded from the streaming session. Instead, it will receive lower priority when it requests streams together with some collaborative peers, which results in a poor quality of service. Every rational peer should behave collaboratively to receive higher quality of streams in the media streaming session.

However, the incentive mechanism also encourages selfish peers to cheat to avoid being identified as selfish. For example,  $P_i$  may create several dummy peers  $P_{i_1}, P_{i_2}, \dots$  and let those dummy peers issue streaming certificates, which can be shown to  $P_i$ ’s upstream peers. This can be prevented because dummy peers will never participate the media streaming session; therefore the streaming certificates issued by them are actually negligible to a collaborative upstream peer. Dummy peers can be further prevented by increasing the cost of getting a valid peer ID in the media streaming system. Another problem is that selfish peers may collude; they can let one of them behave collaboratively to receive media streams that can be forwarded to the rest. However, the possibility is low because they are selfish and none of them would like to fake a collaborative peer by sacrificing its own local resources.

It should be noted that for the peers at the edge of the multicasting tree, they do not have downstream peers and therefore they may be treated as selfish peers. It is not a problem because the multicast tree is highly dynamic as the media streaming process continues and some old peers leave and new peers join. If an edge peer is willing to collaborate, it can find requesting peers to forward streams. Not being an edge peer is exactly an incentive for peers to collaborate.

Our approach is different from some build-in trust and incentive mechanisms in existing P2P file sharing clients. For example, BitTorrent [3] uses a simple method for enforce fair resource sharing, where a peer’s upload rate determines its download rate. However, those mechanisms are enforced by the client itself, therefore they can be circumvented by a skilled user hacking the client and releasing a compatible one without these restrictions. Actually there have been several BitTorrent clients with the capability of adjusting upload and download speed freely. Our approach is robust even if a peer uses a hacked media streaming client, because the incentive mechanism is enforced by the peer’s upstream and downstream peers rather than the peer itself.

The aforementioned incentive mechanism is orthogonal to existing P2P media streaming systems, such as ZIGZAG [12], CoopNet [10], CodedStream [7], PROMISE [5], *etc.* Streaming certificates can be transferred piggyback with the corresponding multicast structure maintenance protocols or the streaming control sequences. Every peer only needs to maintain several records for its upstream and downstream peers. Although we have not tried it out in an available media streaming system, we believe the overhead caused by the

incentive mechanism is negligible.

A similar work [4] proposes a rank-based peer selection mechanism as an incentive to media streaming systems. It is more like a traditional reputation system and focuses on flexible peer selection, whereas our approach pays more attention to monitor and enforce a collaborative behavior in an ongoing media streaming session. We are investigating the possibility of combining the two approaches.

### 3.4 Some Preliminary Results

Here we will give some preliminary experimental results. We simulate a P2P media streaming environment based on the model introduced in Section 2. It should be noted that we do not consider the network topology. The size of the media stream between two peers is determined by their outbound and inbound bandwidth, and the number of other parallel streams. We have 2000 peers that are divided into three types according to their different inbound and outbound bandwidth: 1) 1200 peers have  $B_{in} = 112kbps$  and  $B_{out} = 96kbps$ ; 2) 600 peers have  $B_{in} = 1Mbps$  and  $B_{out} = 600kbps$ ; 3) 200 peers have  $B_{in} = 10Mbps$  and  $B_{out} = 10Mbps$ . There is a media clip which lasts 3600 seconds and has a constant bit rate of  $900kbps$ . The media clip is encoded by MDC into 30 descriptions. We set  $\tau = 30$  seconds. For each media streaming session we randomly choose one peer in the third type as the server peer, and we randomly set various sizes (10%, 20%, 30%) of the peers as selfish ones. We assume all peers participate in the media streaming session. A simple protocol is implemented to construct and maintain multiple multicast trees distributively. We use the average bit rate of the media streams that a peer receives to indicate the quality of service it gets.

When we make the selfish peers to always behave selfishly, the results show that: 1) the average quality of streaming service they receive is less than 1/2 of collaborative ones; 2) most selfish peers do appear at the edge of multicast trees; 3) the average streaming quality received by collaborative peers are not remarkably affected by selfish peers. If we allow the initially selfish peers to behave collaboratively if they can only use less than half of its inbound bandwidth to receive streams at a time, more than 85% selfish peers turn to collaborative peers when the media streaming session is over, and the average bit rate of the media stream received by all peers improves by more than 20%.

## 4. CONCLUSION AND FUTURE WORK

In this paper we briefly introduce an ongoing research work — collaboration-aware P2P media sharing. It combines the traditional reputation system and an online streaming behavior monitoring scheme. The incentive mechanism is orthogonal to streaming solutions and therefore can be integrated into existing P2P media streaming systems. Our preliminary experiments show that the incentive mechanism effectively urges selfish peers to collaborate in a media streaming session. We are currently working on a flow-based network simulator; our next step is to evaluate the proposed mechanism in both the simulator and the PlanetLab [11].

## Acknowledgements

This work is supported by NSF award 0308229. We thank James Ford and Yan Zhao for useful discussions and help with experiments; we also thank the anonymous reviewers

for their comments on earlier drafts.

## 5. REFERENCES

- [1] E. Adar and B.A. Huberman. Free riding on gnutella. *First Monday*, 5(10), October 2000.
- [2] Baruch Awerbuch, Boaz Patt-Shamir, David Peleg, and Mark Tuttle. Collaboration of untrusting peers with changing interests. In *Proceedings of the 5th ACM conference on Electronic commerce*, pages 112–119. ACM Press, 2004.
- [3] Bran Cohen. Incentives build robustness in bittorrent. In *Proceedings of the 1st Workshop on Economics of Peer-to-Peer Systems*, 2003.
- [4] Ahsan Habib and John Chuang. Incentive mechanism for peer-to-peer media streaming. In *Proceedings of International Workshop on Quality of Service 2004 (IWQOS04)*, pages 171–180, June 2004.
- [5] Mohamed Hefeeda, Ahsan Habib, Boyan Botev, Dongyan Xu, and Bharat Bhargava. Promise: peer-to-peer media streaming using collectcast. In *Proceedings of the eleventh ACM international conference on Multimedia*, pages 45–54. ACM Press, 2003.
- [6] Yang hua Chu, Sanjay G. Rao, and Hui Zhang. A case for end system multicast (keynote address). In *Proceedings of the 2000 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, pages 1–12. ACM Press, 2000.
- [7] Baochun Li Jiang Guo, Ying Zhu. Codedstream: Live media streaming with overlay coded multicast. In *Proceedings of the SPIE/ACM Conference on Multimedia Computing and Networking (MMCN 2004)*, pages 28–39, jan 2004.
- [8] Maxwell N. Krohn, Michael J. Freedman, and David Mazières. On-the-fly verification of rateless erasure codes for efficient content distribution. In *Proceedings of IEEE Security and Privacy 2004*, pages 226–240.
- [9] Sergio Marti and Hector Garcia-Molina. Limited reputation sharing in p2p systems. In *Proceedings of the 5th ACM conference on Electronic commerce*, pages 91–101. ACM Press, 2004.
- [10] Venkata N. Padmanabhan, Helen J. Wang, Philip A. Chou, and Kunwadee Sripanidkulchai. Distributing streaming media content using cooperative networking. In *Proceedings of the 12th international workshop on Network and operating systems support for digital audio and video*, pages 177–186. ACM Press, 2002.
- [11] PlanetLab. <http://www.planet-lab.org>.
- [12] Duc A. Tran, Kien A. Hua, and Tai T. Do. A peer-to-peer architecture for media streaming. *IEEE Journal on Selected Areas in Communications*, 22(1):121–133, 2004.
- [13] Dongyan Xu, Mohamed Hefeeda, Susanne Hambrusch, and Bharat Bhargava. On peer-to-peer media streaming. In *Proceedings of the 22nd International Conference on Distributed Computing Systems (ICDCS'02)*, page 363. IEEE Computer Society, 2002.
- [14] Wenwu Zhu Zhensheng Zhang Ya-Qin Zhang Zhe Xiang, Qian Zhang. Peer-to-peer based

multimedia distribution service. *IEEE Transaction on  
Multimedia*, 6(2):343–355, April 2004.