P2P VIDEO LIVE STREAMING WITH MDC: PROVIDING INCENTIVES FOR REDISTRIBUTION

Zhengye Liu, Yanming Shen, Shivendra S. Panwar, Keith W. Ross, Yao Wang

Polytechnic University, Brooklyn, NY11201, USA
Email: {zliu04,yshen02}@utopia.poly.edu, {panwar,ross,yao}@poly.edu

ABSTRACT

In this paper, we consider applying multiple description coding in data-driven P2P live streaming networks to provide incentives for redistribution. In our system, a video is encoded into multiple descriptions with each description having equal importance. We consider a heterogeneous system with peers having different uplink bandwidths. We design a distributed protocol in which a peer contributing more uplink bandwidth receives more descriptions and consequently better video quality. Previous approaches consider single-layer video, where each peer receives the same video quality no matter how much bandwidth it contributes to the system. The simulation results show that our approach can provide differentiated video quality commensurate with a peer's contribution to other peers.

1. INTRODUCTION

P2P video live streaming has become an extremely popular service in the Internet. Several streaming systems have been successfully deployed, serving tens of thousands of simultaneous users who watch channels at rates between 300 kbps to 1Mbps [1, 2, 3, 4].

In P2P streaming systems, participating peers have different upload bandwidths. Institutional peers have high-bandwidth access, while residential peers with DSL and cable access have relatively low upload bandwidths. A high bandwidth user uploads significantly more content than a low bandwidth user. In [4], it has been reported some institutional peers contribute 30 times more uplink bandwidth than residential peers. However, most existing systems only adopt single layer video coding, where each peer receives the same video quality no matter how much bandwidth it contributes to the system. This may create a serious free-rider problem, similar to what has been observed in P2P file sharing systems[5]. Preferably, a peer should receive a better video quality if it contributes more bandwidth. This will encourage peers to contribute more uplink bandwidth to the system and prevent free-riders.

To provide incentives for redistribution, in this paper, we consider applying multiple description coding (MDC) techniques on a data-driven P2P video live streaming system. With MDC, peers receiving different number of descriptions will have different video quality, with more received descriptions introducing better video quality. In this paper, we propose a distributed protocol, in which a peer contributing more uplink bandwidth is more likely to receive more descriptions and consequently a better video quality. Therefore, peers are encouraged to be cooperative. The simulation results show that our approach can provide differentiated video quality commensurate with a peer's contribution to other peers, while maintaining a high overall system performance.

The rest of the paper is organized as follows. In section 2, we describe our system and its design. Extensive simulation results are presented in Section 3, and Section 4 concludes the paper.

2. SYSTEM ARCHITECTURE AND DESIGN

The data-driven delivery architecture for video live streaming bears strong similarities to BitTorrent [6]. Most successful P2P video streaming systems [1, 2, 3] adopt this architecture. In such a system, a video is divided into media chunks, and the chunks are made available at an origin server. When a peer wants to view the video, it obtains a list of peers currently watching the video and establishes partnership with several randomly selected peers. Peers exchange chunk availability information with their neighbors and request chunks that they need. This data-driven approach is particularly desirable for the highly dynamic, high-churn P2P environment [3].

To provide redistribution incentives and enable differentiated service among peers, a natural approach is to apply scalable video coding. With our approach, a peer that contributes more can receive a better video quality, while a peer that contributes less may receive a worse but acceptable video quality. In our design, we use MDC [7] encoded video. With MDC, the origin server encodes a video into M descriptions. Each description is further divided into description chunks (DC) of Δ seconds, then peers exchange the DCs. The main reason of adopting MDC is that MDC improves the content availability in a P2P video streaming system. As illustrated in Fig. 1, with MDC, it is always possible for a high contribution peer to receive a large number of descriptions from its neighbors, even though all the neighbors are holding a small number of

descriptions. Therefore, with an appropriate design, the uplink bandwidth contribution is the only factor that affects the video quality of a peer. This simplifies the system design.

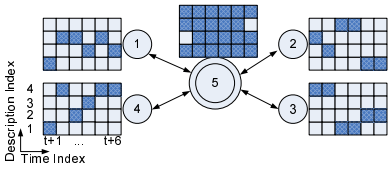


Fig. 1. MDC in P2P streaming with differentiated service. The circles indicate peers, and the rectangles indicate the buffer of each peer. A shaded rectangle represents that the chunk is available. Peers 1, 2, 3, and 4 contribute less and consequently are supposed to receive one description. Peer 5 is a peer with high uplink bandwidth and is supposed to receive four descriptions. With MDC, even if all the neighbors of peer 5 only hold one description, peer 5 can receive a large number of descriptions.

Our proposed scheme uses an idea similar to the tit-for-tat[8] strategy, which has been successfully used in BitTorrent. With the tit-for-tat strategy, *a peer rewards more to a neighbor who contributes more to it*. In our scheme, each peer measures its download rate from its neighbors. A peer reciprocates to its neighbors by providing a larger fraction of its upload rate to the neighbors from which it is downloading at the highest rates. A peer with a higher uplink bandwidth contribution is more likely to obtain higher priorities in more neighbors, thus receiving a better video quality.

Our proposed strategy applies both supplier side scheduling and receiver side scheduling. As a supplier, a peer may receive multiple DC requests from its neighbors. The peer should determine which request should be served first. As a receiver, in general, a peer has a choice of several DCs that it can download. For a particular DC, there may be several neighbors holding it. Thus, the peer has to decide which DC to request first and from which neighbor to request it. In this paper, we assume there is no downlink bandwidth constraint.

2.1. Supplier Side Scheduler

In our design, a peer will upload more to the neighbor from which this peer downloads more. To this end, a supplier maintains a different request queue for each receiver. For a particular receiver, the queue is first-in-first-out, where the supplier serves the requests in the order that the requests were received. The supplier transmits one requested DC to one receiver at one time. The supplier determines which receiver should be served depending on the receiver's contribution to the supplier. At any one time, the supplier randomly selects a receiver to serve. Let p_k denote the probability that peer n selects the receiver k. p_k is determined as follows:

$$p_k = \frac{S_k d_{n,k}}{\sum_{i=1}^{N} S_k d_{n,k}}, \tag{1}$$

where N is the number of neighbors of peer n, $d_{n,k}$ is the estimated download rate of supplier n from receiver k. $d_{n,k}$ can be obtained based on the number of DCs delivered from peer k to peer n during the previous time period. S_k equals to 0 if the request queue of receiver k is empty, otherwise, S_k equals to 1. Therefore, a supplier serves its receivers in a weighted fashion, and a receiver that uploads more to the supplier has a higher probability of being served.

2.2. Receiver Side Scheduling

At any given time, a receiver will have buffered DCs that are to be displayed in the future. Figure 2(a) shows an example for the DCs buffered at a particular receiver, where the X's denote buffered DCs. In this example, there are 8 DCs buffered: 3 for the next DC time, 2 for the second DC time, and so on. Other DCs have been requested but have not yet arrived. These are shown with dashes in Fig. 2(a). In this example, there are three such requested-but-yet-to-arrive DCs. There are DCs that are available in the neighbors, but have not been requested. These DCs are left blank in Fig. 2(a). Additionally, a non-buffered DC may not be available from any of the neighboring peers. These are shown by circles in Fig. 2(a). These buffered DCs, requested DCs, available-but-yet-to-request DCs, and not-available DCs constitute the current *buffer state* of the receiver.

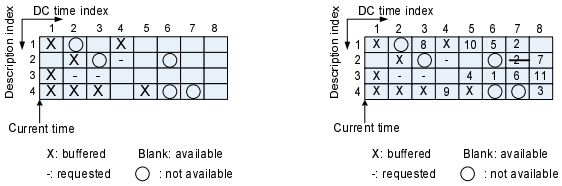


Fig. 2. Buffer state at a given time and the sequence that chunks are requested.

In our design, peers request DCs at the beginnings of rounds. At the beginning of each round, the peer will have a buffer state, as described above. Given this buffer state, the peer must decide which DCs to request. The peer may request DCs from the current time up until a window of B DC times into the future. Figure 2(a) uses a window of $B = 8$.

In requesting DCs, there are two conflicting goals. On one hand, we want to have as many descriptions as possible that can be played out in the near future. On the other hand, we want to ensure that there is not a lot variation in the number of descriptions that are displayed from one DC time to the next.

We use a simple but natural heuristic for prioritizing the DCs that are to be requested at the beginning of a round. We first look for the first DC time (scanning from the current time towards the end of the window) that has no buffered or requested DCs. Assuming there is such a DC time, we randomly choose a DC from those that are available. Figure 2(b) shows how DCs are requested, where the number in each box represents the sequence that the DC is requested. In Fig. 2(b), the first DC time that has no buffered or requested DCs is the sixth DC time. For this DC time there are two available DCs. We randomly select one of these DCs, which becomes the DC

that is requested first. After selecting the DC, it is possible that more than one neighbor may have this DC. The receiver will send the request to a neighbor, from which the DC is expected to be received at the earliest time. Continuing with the heuristic, the first DC time with no buffered or requested DCs is now DC time 7; for this DC time, there are four available DCs, and we can choose one of those DCs at random. We continue with this heuristic until every DC time has at least one buffered or requested DC (neglecting DC times that have no available DCs, that is, all circles). We then continue with heuristic, giving priority to DC times that have exactly one buffered or requested DC.

A receiver k can estimate the time $\tau_{k,n}$ that a DC will be received from its neighbor peer n by the following equation:

$$\tau_{k,n} = R_{k,n}\frac{r\Delta}{d_{k,n}}, \tag{2}$$

where $R_{k,n}$ is the number of outstanding requests from k to n, r is the bitrate of one description, and $d_{k,n}$ is the estimated download rate of peer k from peer n. Note that in our heuristic, a peer will send DC request to a neighbor that can deliver this DC at the earliest time. Therefore, if $\tau_{k,n}$ is the minimum, and if it is less than this DC's playback deadline, then peer k will send the request to peer n. If this DC cannot meet its playback deadline, it will not be requested, and the heuristic will go to the next DC. In Fig. 2(b), the "2" with a dash at the seventh DC time means that this DC should be the second DC to be requested with the given buffer state. However, this DC cannot meet its playback deadline based on the estimation in eqn. (2) for all the neighboring peers of receiver k. Thus the receiver will not send out request for this DC and select another DC at DC time 7 instead.

Given the total potential download rate of a peer, which depends on its upload contribution, if a peer requests too aggressively for DCs with a particular DC time and receives a video rate higher than its potential download rate at that time, then this peer is more likely to receive a lower video rate at other times. This will lead to a video quality variation at the receiver. In this design, to deduce quality variation, a peer k maintains a threshold Γ_k, which is set to half between the current download rate D_k and the potential download rate U_k, plus an aggresivity factor α,

$$\Gamma_k = \lfloor \frac{D_k + U_k}{2r} + \alpha \rfloor, \tag{3}$$

where r is the bitrate of one description. At a given DC time, if there are more than Γ_k DCs that have already been buffered or requested, peer k will not request any additional DC at that time. When D_k is much less than U_k, D_k is more likely to or has more room to increase. Thus the peer requests more aggressively. After receiving the requested DCs, a peer with a high U_k can potentially contribute more to its neighbors. In turn, this improves the probability that this peer can be served by its neighbors. Therefore, it leads to an increased download bitrate for peer k. The fast-start phase accelerates the convergence process from a lower D_k to the target receiving rate $D_k = U_k$. When D_k reaches U_k, a peer requests $\lfloor \frac{U_k}{r} + \alpha \rfloor$ descriptions, which is comparable to its uplink bandwidth contribution. As shown in Fig. 2(b), where $\Gamma_k = 3$, there are at most 3 DCs that have been buffered or requested for each DC time, no matter how many DCs are available at that DC time.

Like BitTorrent, in order to locate a better neighbor, which has higher uplink bandwidth, in our scheme a peer periodically replaces the neighbor with the least contribution by a randomly selected peer. The peer initially assumes that the newly joined neighbor is cooperative and has a reasonable uplink bandwidth. After several rounds, the download rate from the new neighbor can be evaluated.

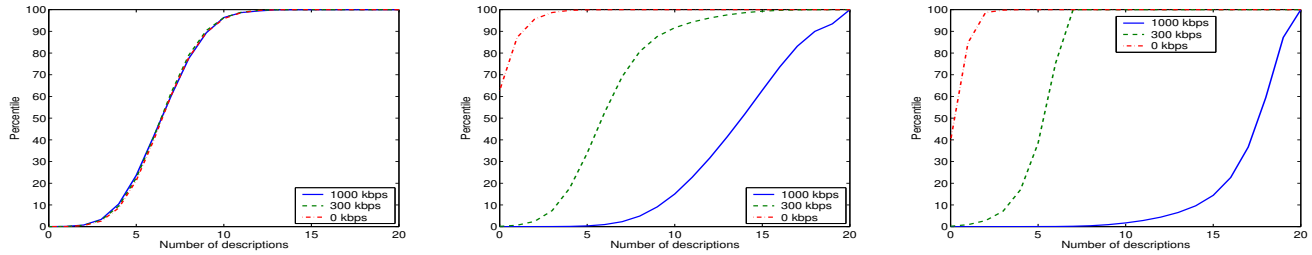
3. PERFORMANCE STUDIES

In our simulation, a video is totally encoded into 20 descriptions at the origin server. Each description is further divided into one second period DCs. The bitrate of each description is 50 kbps. There are 1000 peers in the overlay, and the requesting round is one second. The peers are classified into three types: 10% of the peers are Ethernet users with 1000 kbps uplink bandwidth, 85% of the peers are cable users with 300 kbps uplink bandwidth, and 5% of the peers are free-riders.

The buffer length is set to 30 seconds. Each peer estimates a neighbor's contribution based on the number of DCs received from it in the previous 60 seconds. Initially, each peer has 14 neighbors. Peer k will accept the partnership request from peer n if peer k has less than 18 neighbors; otherwise peer k will reject the request. Currently, if peer k has larger than 14 neighbors, when it drops its worst neighbor, it will not search a new neighbor. When peer k and peer n newly establish a partnership, they assume the bandwidth allocated from each other is 10 kbps. The total simulation time is set to 3600 seconds. At the initial stage of the simulation, each peer does not have any knowledge about its neighbors, and peers are randomly grouped. During every 30 second period, a peer replaces the neighbor with the least contribution by a randomly chosen peer.

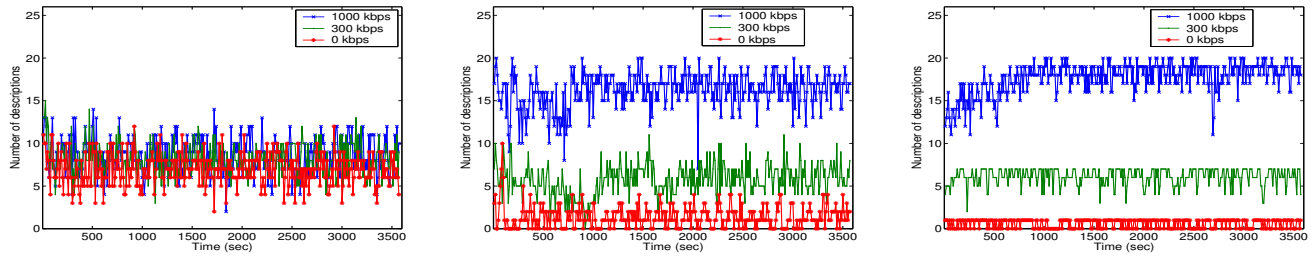
We compare three strategies: (i) without differentiated service, where a supplier ignores the contribution of the receivers and serves the requests from different receivers in a round-robin order; (ii) differentiated service with greedy requests, where the threshold Γ_k in eqn. (3) is not applied.; (iii) differentiated service with adaptive requests, where Γ_k in eqn. (3) is applied.

Figure 3 plots the cumulative distribution of the number of the received descriptions for three types of users with different strategies. We record the number of the received descriptions of each peer in each one second time slot. From Fig. 3, we can see without the weighted round-robin scheduling at a supplier, all peers receive similar video quality, thus the CDFs of Ethernet users, cable users and free-riders are indistinguishable. It is obviously unfair that free-riders and Ethernet users receive the same video quality. Figure 3(b)



(a) (b) (c)

Fig. 3. Cumulative distribution of number of received descriptions for all peers during the simulation period. (a) without differentiated service; (b) Differentiated service with greedy requests; (c) Differentiated service with adaptive requests.



(a) (b) (c)

Fig. 4. Behavior of typical peers. (a)Without differentiated service; (b) Differentiated service with greedy requests; (c) Differentiated service with adaptive requests.

and Figure 3(c) show the CDFs when differentiated service is enabled. We can see that the CDFs of different types of users are separated. Free-riders receive a very poor video quality, without receiving a single description for over 40% of the time slots, which is unacceptable for video service. Our proposed strategy can virtually eliminate free-riders from the system. Both cable users and Ethernet users can receive the bitrates comparable to their uplink bandwidth contribution. Figure 3(c) shows the CDFs for the adaptive requesting strategy. For each type of users, the number of received descriptions has a narrow range, thus indicates a more stable download rate for each user. We also observe that the system aggregate throughput under the adaptive request scheme (93%) is slightly lower than the other two schemes (both are 97%). To pursue a stable video quality, a peer requests the DCs more conservatively, so that there is a slight sacrifice in throughput.

Figure 4 shows the behavior of typical users with three strategies. We randomly chose three peers, each from one user category, and sample their number of received descriptions per 10 seconds. As shown in Fig. 4(a), without enabling differentiated service, each peer receives similar video qualities. In Fig. 4(b), it is easy to differentiate between the Ethernet user and the cable user. However, the number of received descriptions changes dramatically for different times, which introduces an unstable video quality. In Fig. 4(c), the fluctuation is significantly reduced. Note that in our simulation, at the initial state, each peer in the overlay does not have any knowledge about its neighbors. As shown in Fig. 4(c), even with this extreme case, the download rate of a peer can gradually converge to a bitrate which is comparable to its uplink bandwidth contribution.

4. CONCLUSION

In this paper, we propose a distributed solution which combines the MDC with the data-driven approach to provide redistribution incentives and enable differentiated service in P2P live streaming. A tit-for-tat like strategy is proposed. With our design, a peer contributing more uplink bandwidth will receive a better video quality. Therefore, peers are encouraged to contribute more uplink bandwidth.

5. REFERENCES

[1] "PPLive." [Online]. Available: http://www.pplive.com/

[2] "PPStream." [Online]. Available: http://www.ppstream.com/

[3] X. Zhang, J. Liu, B. Li, and P. Yum, "DONet: A data-driven overlay network for efficient live media streaming," in *Proc. of IEEE INFOCOM*, 2005.

[4] X. Hei, C. Liang, Y. Liu, and K. W. Ross, "Insights into PPLive: A measurement study of a large-scale P2P IPTV system," in *Workshop on Internet Protocol TV (IPTV) services over World Wide Web*, Edinburgh, Scotland, May 2006.

[5] P. Colle, K. Leyton-Brown, and I. Mironov, "Incentives for sharing in peer-to-peer networks," in *ACM Conference on Electronic Commerce 2001*, 2001.

[6] "BitTorrent." [Online]. Available: http://www.bittorrent.com/

[7] Y. Wang, A. R. Reibman, and S. Lin, "Multiple description coding for video delivery," *Proceeding of the IEEE*, pp. 57–70, Jan. 2005.

[8] B. Cohen, "Incentives build robustness in BitTorrent," in *1st Workshop on Economics of Peer-to-Peer Systems*, June 2003.