IEICE TRANS. INF. & SYST., VOL.E100–D, NO.3 MARCH 2017

LETTER

A Hybrid Push/Pull Streaming Scheme Using Interval Caching in P2P VOD Systems

Eunsam KIM[†], **Boa KANG**[†], ***Nonmembers***, ***and*** **Choonhwa LEE**[††a)], ***Member***

SUMMARY This paper presents a hybrid push/pull streaming scheme to take advantage of both the interval caching-based push method and the mesh-based pull method. When a new peer joins, a mesh-based pull method is adopted to avoid the overhead to reorganize the structure only if all of its potential preceding peers are likely to leave before the end of its playback. Otherwise, an interval caching-based push method is adopted so that the better performance of the push method can be maintained until it completes the playback. We demonstrate that our proposed scheme outperforms compared with when either the interval caching-based push method or mesh-based pull method is employed alone.

key words: *P2P streaming, VOD systems, interval caching, push/pull transmission method*

1. Introduction

Recent advances in computer networks and digital technology have enabled the development of personalized multimedia services such as VOD (Video On Demand). However, existing client/server architectures based on CDN (Content Distribution Networks) have been limited in scalability because of the high installation costs when the number of users increases. To address this problem, many research efforts have been made regarding peer-to-peer (P2P) streaming due to its high scalability and low installation cost [1], [2].

In P2P VOD systems, data transmission methods have been broadly classified into push and pull. On the other hand, interval caching is known to be an effective buffering scheme that can improve the performance in VOD systems [3]–[5]. By caching data in the interval between two successive streams for the same video after forming a pair of preceding-following peers, this scheme can serve the following stream without any actual disk accesses in the storage systems. When the interval caching is employed in P2P VOD systems, a preceding peer must cache all the data for the interval in its memory. This enables the preceding peer to send data to its following peer in a push method. This means that the following peer only needs to issue one request initially for all subsequent chunks. As a result, the interval caching-based push method requires a relatively small number of control messages and involves short delay and no redundancy of transmitted data. However, this method requires considerable overhead to rebuild the structure whenever each peer joins or leaves. It is also evident that the following peers cannot utilize their upload bandwidth since they always receive data from the preceding peers and do not send any data.

On the other hand, in the pull method, peers exchange buffermaps with each other to identify which peer stores the required chunks [6], [7]. Since each peer exchanges data with multiple neighbor peers at the same time, this method provides a robust structure against peers' churn and enables the bandwidths of all peers to be utilized. However, this method creates long startup delay since peers cannot receive data sequentially like in the push method and requires a large number of data exchanges for the explicit request of each required chunk. In addition, data redundancy occurs since the same chunks can be received from multiple peers.

In this paper, we propose a hybrid push/pull streaming scheme to take benefit of both the interval caching-based push method and the mesh-based pull method simultaneously. Note that the push method shows better performance except that it requires considerable overhead to rebuild the structure when peers leave and the following peers cannot utilize their upload bandwidth. In our proposed scheme, when a new peer joins, we therefore adopt a mesh-based pull method to avoid the overhead to reorganize the structure only if all of its potential preceding peers are likely to leave before the end of its playback. Otherwise, an interval caching-based push method is adopted so that the superior performance of the push method can be maintained until it completes the playback. This is possible because at least one preceding peer is unlikely to leave in this case. Furthermore, the problem of the following peers being unable to utilize their upload bandwidth is also resolved since they can participate as neighbor peers in the mesh method.

Through extensive simulations, we demonstrate that the ratio of data received from the server in our proposed hybrid push/pull streaming scheme is much lower compared with when either the interval caching-based push method or mesh-based pull method is employed alone.

The present paper is organized as follows: Sect. 2 describes work related to this paper. Section 3 describes our proposed hybrid push/pull streaming scheme in detail. Section 4 presents the experimental results of the hybrid push/pull scheme in comparison with those of an interval caching-based push method and mesh-based pull method. Finally, Sect. 5 offers conclusions.

Manuscript received September 5, 2016.
Manuscript revised November 14, 2016.
Manuscript publicized December 6, 2016.

[†]The author are with the Dept. of Computer Engineering, Hongik University, Seoul, 121–791 Korea.
[††]The author is with the Division of Computer Science and Engineering, Hanyang University, Seoul, 133–791 Korea.
a) E-mail: lee@hangyang.ac.kr

DOI: 10.1587/transinf.2016EDL8183

2. Related Work

Interval caching is a typical buffering scheme used in VOD server systems [3]. This scheme sorts out the intervals between successive requests according to their lengths and caches the intervals whose lengths are shorter first until the buffer space is filled. By doing so, the VOD server systems can utilize a given cache space as efficiently as possible. Accordingly, many studies on the extension of the interval caching scheme have been conducted. A popularity-aware interval caching scheme that predicts future requests via prior request frequency has been proposed [4]. The distributed interval caching scheme, which uses in-disk buffers to store intervals in network-attached disk structures, has also been proposed [5]. However, most studies have aimed to reduce disk I/O operations in VOD servers rather than in P2P structures.

On the other hand, research exploiting the interval in P2P VOD systems has been conducted. In DirectStream, clusters are formed as a tree structure based on the joining time of each peer [8]. Only a root peer in each of the clusters receives data from the server, while the other peers in each cluster are connected in a chain from its root so that following peers can receive data from preceding peers in the push method. If a peer leaves, all of its child peers may receive data from the server since it is based on a tree structure. As the peer is closer to a root, the impact of the peer's leaving worsens. To avoid such problems of tree structures, many P2P streaming systems based on mesh structures have also been proposed [6], [7]. However, they create long startup delay and require a large number of exchanges of control messages among peers.

To provide a good tradeoff between two structures, several hybrid push/pull methods have therefore been proposed [9]–[11] but they focus mainly on the integration of the tree and mesh structures. In particular, P2VoD forms an additional mesh-based structure called generation on top of application-layer multicast trees to avoid the performance degradation when peers leave, which makes the system more complicated to maintain them [11]. Moreover, in P2VoD, any leaf peers cannot utilize their upload bandwidth since the system basically operates based on tree structures. Our proposed scheme is different from P2VoD in that each peer uses only one of two structures depending on the probability that all of its potential preceding peers leave during its playback. In addition, peers can fully utilize their upload bandwidth since they participate as neighbor peers in mesh structures. Note that, to the best of our knowledge, no hybrid methods have tried to exploit the interval caching technique to maximize the effect of the integration of both structures.

3. A Hybrid Push/Pull Streaming Scheme

A P2P VOD system consists of a media server, a tracker server, and peers. A media server stores and manages all video data. If the playback intervals between peers become longer or a certain level of playback quality is not guaranteed due to a small number of neighbor peers, a media server transmits data directly to corresponding peers. A tracker server manages information about currently connected peers. It also manages cached interval information (i.e. pairs of a preceding and a following peer and their interval lengths).

3.1 Peer Joining

Once a new peer joins, the tracker server decides on a data transmission method, choosing either an interval caching-based push method or a mesh-based pull method. First, the tracker server evaluates the number of preceding peers that can send the required data to the new peer in an interval caching-based push method (denoted by N_p). That is, it calculates the number of preceding peers each of whose buffer ranges includes the current playback position of the new peer.

If the probability that at least one among all potential preceding peers of the new peer does not leave until its playback is completed is high, the interval caching-based push method is chosen. This indicates that the system can maintain better performance of the push method as well as reduce the overhead for rebuilding the structure by minimizing the possibility that a peer will leave. Otherwise, the mesh-based pull method is chosen because it is more robust against a peer's leaving.

To determine the criteria for choosing the transmission method, the tracker server maintains joining and leaving times of all peers. Each peer sends messages to the tracker server when it joins and leaves. In particular, to handle the situation where peers leave unexpectedly due to catastrophic events such as computer crashes, peers periodically exchange keep-alive messages with each other. When a peer has not received any keep-alive message from one of its neighbor peers for a given timeout duration, it informs the tracker server that the neighbor has left.

From these data, the tracker server can calculate peer leaving rate during a predetermined time period (denoted by λ_l) by dividing the total number of peers that have left by the time period in seconds. It can also evaluate the average playback duration of all peers (denoted by d). Thus, we can obtain the expected number of peers that will leave during d by computing $\lambda_l \times d$. If $N(t)$ represents the total number of participating peers at time t when a peer joins, the expected ratio of peers that will leave during d from time t (denoted by r_l) can be obtained with Eq. (1).

$$r_l = \frac{\lambda_l \times d}{N(t)} \tag{1}$$

It is obvious from Eq. (1) that the expected ratio of peers that will stay during d from time t (denoted by r_s) becomes $1 - r_l$ as shown in Eq. (2).

$$r_s = 1 - r_l \tag{2}$$

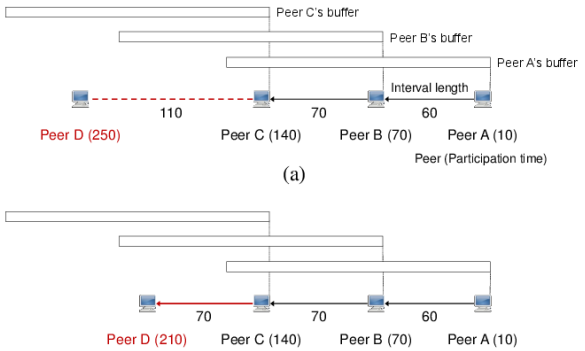

Fig. 1 An operation example when a new peer joins. (a) In case when a new peer D joins at 250 seconds. (b) In case when a new peer D joins at 210 seconds.

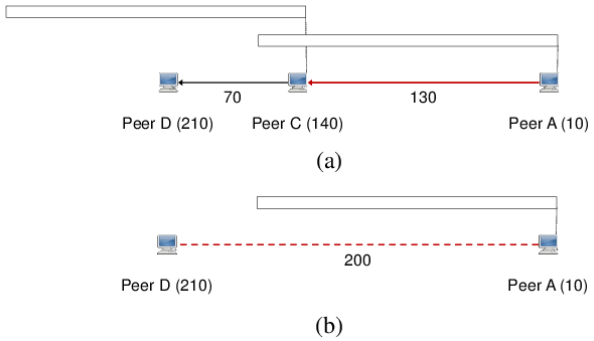

Fig. 2 An operation example when a peer leaves. (a) In case when peer B leaves. (b) In case when peer C leaves.

Therefore, when a new peer joins at time t and N_p peers are participating, $N_p \times r_s$ peers will be expected to stay in the system during d. To choose the interval caching-based push method, at least one among all potential preceding peers of the new peer must stay. That is, if $N_p \times r_s$ is equal to or greater than one as shown in Eq. (3), we choose the interval caching-based push method. Otherwise, we choose the mesh-based pull method.

$$N_p \times r_s \geq 1 \tag{3}$$

Figure 1 shows an operation example when peers leave at 0.5 of r_l in the P2P VOD system. Figure 1 (a) shows a case where a new peer D joins at 250 seconds. In this case, N_p is one and r_s is 0.5. Since $N_p \times r_s$ is only 0.5, the condition for choosing an interval caching-based push method, i.e., Eq. (3), is not satisfied. Thus, a mesh-based pull method is chosen. On the other hand, if a new peer D joins at 210 seconds as shown in Fig. 1 (b), N_p is two. Since $N_p \times r_s$ is one, an interval caching-based push method is chosen.

3.2 Peer Leaving

Once a preceding peer leaves in an interval caching-based push method, its following peer tries to find a new preceding peer on a chain of successive pairs of preceding/following peers so that it can continue to take advantage of the interval caching-based push method. However, if an appropriate preceding peer cannot be found, a mesh-based pull method is employed.

Figure 2 (a) shows an operation example when peer B leaves in Fig. 1 (b). In this case, peer C becomes disconnected from peer B, which is its preceding peer. A tracker server then checks which peer can be its preceding peer by using the cached interval information. We can see that peer A can be its preceding peer since the current playback position of peer C is included within the buffer range of peer A. Accordingly, peer C can receive data from peer A while continuing to use the interval caching-based push method.

When peer C leaves, as shown in Fig. 2 (b), the tracker server also checks whether any preceding peer exists for peer D in the same manner. Since the current playback position of peer D is not within the buffer range of peer A, however, the interval caching-based push method cannot be used any longer. Thus, peer D must receive data in a mesh-based pull method after it is connected to multiple neighbor peers.

It is noted that, in order to guarantee the minimum playback quality in the mesh-based pull method, a peer must receive data directly from the media server if it cannot be connected to more than a certain number of neighbor peers.

4. Performance Evaluation

To show the effectiveness of our proposed hybrid push/pull streaming scheme, we have performed extensive simulations using the PeerSim P2P simulator. The bandwidth between the general peer and router is set to an evenly dispersed range of 10–100 Mbps, and the backbone bandwidth is set to 10 Gbps. Each video has a 720-Kbps playback rate and the video data is divided at a rate of 3 chunks per second. The 512 chunks can be stored in the buffer of each peer. To show the effect of interval length between peer arrivals on the system performance, we vary the average peer inter-arrival times between 10 sec., 30 sec., 60 sec., and 100 sec. To examine the impact of the ratio of peers that will leave during a specific period, we also changed the ratio of leaving peers (i.e., r_l) between 0.2, 0.4, and 0.6. To set r_l to these values, we varied λ_l between 0.04, 0.08 and 0.12 while fixing d and $N(t)$ to 500 sec. and 100, respectively.

When an interval caching-based push method is employed alone, a peer has to receive data directly from the media server when it cannot find a preceding peer. When a mesh-based pull method is employed alone, the peer must also receive data from the server if the number of neighbor peers that it can be connected to is less than four because a certain level of playback quality cannot be guaranteed in this case. Note that the maximum number of neighbor peers is limited to six in the mesh-based pull method throughout our simulations.

We compare the performance of the following three cases as shown in Fig. 3: (i) where an interval caching-based push method is employed alone (IC_PS), (ii) where an exist-

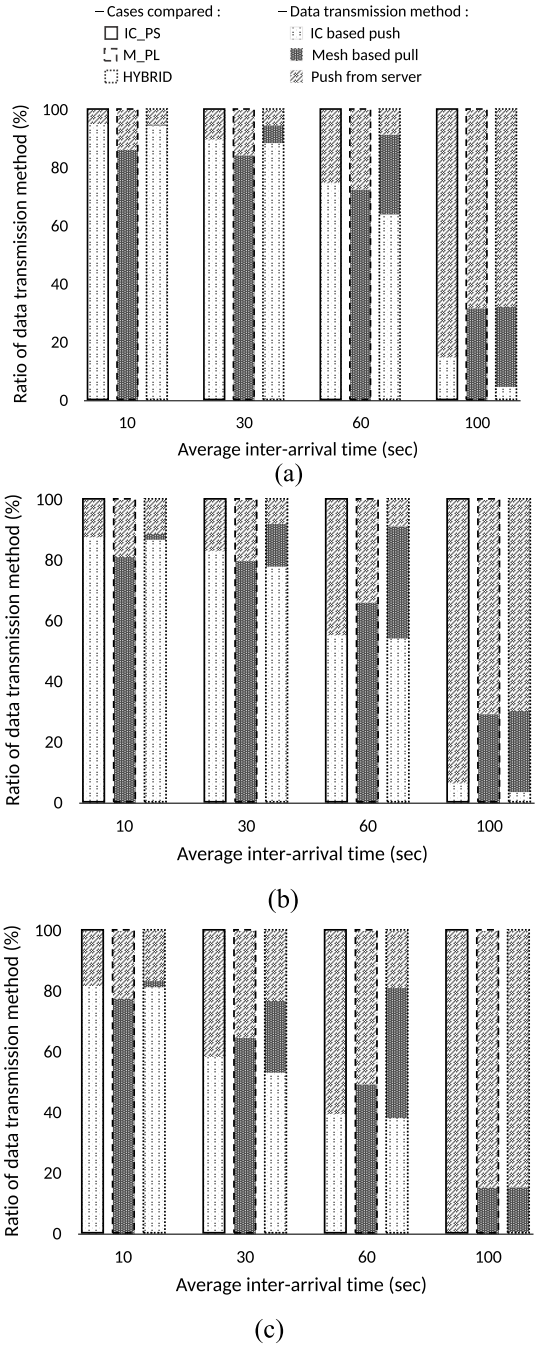

Fig. 3 Comparison of data transmission methods according to inter-arrival time between peers. (a) Average ratio of leaving peers is 0.2, (b) average ratio of leaving peers is 0.4, and (c) average ratio of leaving peers is 0.6.

ing mesh-based pull method is employed alone (M_PL), and (iii) where our proposed hybrid push/pull streaming scheme is employed (HYBRID). We evaluate the performance of each case in terms of the ratio of data received by the following three transmission methods: an interval caching-based push method, a mesh-based pull method, and a push method directly from the media server. Note that the goal of P2P streaming is to minimize the server load by reducing the direct data transmission from the server as much as possible.

We first compare the following two cases: IC_PS and M_PL. We can see from Fig. 3 that the ratio of data received from the server in IC_PS is relatively lower than that in M_PL as either the ratio of leaving peers or inter-arrival time decreases. The ratio of data received from the server in IC_PS is on average 5.1% lower than that in M_PL when the pairs of average ratio of leaving peers and inter-arrival time have the following values: (0.2, 10 sec.), (0.2, 30 sec.), (0.2, 60 sec.), (0.4, 10 sec.), (0.4, 30 sec.), and (0.6, 10 sec.) In particular, in the case of (0.2, 10 sec.), the ratio in IC_PS is 9% lower than that in M_PL. This is mainly because IC_PS outperforms M_PL as long as peers do not leave, as mentioned above. Another reason is that, even though a preceding peer leaves in these situations, its following peer is more likely to find its preceding peers as the peer inter-arrival time decreases. That is, the playback position of the preceding peer still tends to remain in the buffer range of other peers.

Conversely, in the other situations, (i.e., where either the ratio of leaving peers is higher or the inter-arrival time is longer), the ratio of data received from the server in M_PL is on average 13.5% lower than that in IC_PS. The reason for this is that the mesh structure is more stable when peers leave because each peer is connected to at least four neighbor peers. When a ratio of leaving peers is high in IC_PS, however, it is harder to find its new preceding peer. Similarly, as the peer inter-arrival time increases in IC_PS, the probability that the buffer ranges of peers are overlapped with each other becomes lower.

On the other hand, it can be seen from Fig. 3 that HYBRID achieves the lowest ratio of data received from the server in all ratios of leaving peers and peer inter-arrival times. That is, the ratios in HYBRID are 15.4% and 11.2% lower than those in IC_PS and M_PL, respectively. The main reason for this is that the system can significantly reduce the overhead caused by peers' leaving by choosing IC_PS only when at least one among all potential preceding peers is likely to remain until the end of the playback. Another reason for such performance improvement is that the system can increase the data availability by deciding to choose M_PL which is robust against peers' leaving when all potential preceding peers are likely to leave.

The experimental results thus indicate that HYBRID can decrease the ratio of data received from the server considerably compared with when either IC_PS or M_PL is employed alone by fully utilizing the advantages of both IC_PS and M_PL (i.e. better performance and robustness against peers' churn). It is also noted that the upload bandwidths of all peers can be utilized to exchange data since all peers can participate as neighbor peers in M_PL.

5. Conclusions

In this paper, we proposed a hybrid push/pull streaming scheme in P2P VOD systems. When a new peer joins, by adopting an interval caching-based push method only if at least one among its potential preceding peers is likely to remain before the end of its playback, we can take advantage

of better performance of the push method, Otherwise, we can construct the more robust structure against peers' churn by adopting a mesh-based pull method. The simulation results showed that our proposed scheme performs better than other existing methods.

Acknowledgments

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2016R1D1A1A09917396) and by the MSIP (Ministry of Science, ICT&Future Planning), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2016-H8501-16-1006) supervised by the IITP (Institute for Information & communications Technology Promotion).

References

[1] D. Kim, E. Kim, and C. Lee, "Efficient peer-to-peer overlay networks for mobile IPTV services," IEEE Trans. Consum. Electron., vol.56, no.4, pp.2303–2309, 2010.

[2] C. Lee and E. Kim, "Boosting P2P streaming performance via adaptive chunk selection," IEICE Trans. Commun., vol.E94-B, no.10, pp.2755–2758, Oct. 2011.

[3] A. Dan and D. Sitaram, "Generalized interval caching policy for mixed interactive and long video workloads," Proc. SPIE - Multimedia Computing and Networking, pp.344–351, 1996.

[4] O. Kwon, H. Bahn, and K. Koh, "Popularity and prefix aware interval caching for multimedia streaming servers," Proc. IEEE Conf. on Computer and Information Technology, pp.555–560, 2008.

[5] N.J Sarhan and C.R. Das, "Caching and scheduling in NAD-based multimedia servers," IEEE Trans. Parallel Distrib. Syst., vol.15, no.10, pp.921–933, 2004.

[6] X. Hei, C. Liang, J. Liang, Y. Liu, and K.W. Ross, "A measurement study of a large-scale P2P IPTV system," IEEE Trans. Multimedia, vol.9, no.8, pp.1672–1687, 2007.

[7] X. Zhang, J. Liu, B. Li, and Y.-S.P. Yum, "CoolStreaming/DONet: A data-driven overlay network for peer-to-peer live media streaming," Proc. INFOCOM, pp.2102–2111, 2005.

[8] Y. Guo, K. Suh, J. Kurose, and D. Towsley, "DirectStream: A directory-based peer-to-peer video streaming service," Computer Communications, vol.31, no.3, pp.520–536, 2008.

[9] Y. Tang, J.-G. Luo, Q. Zhang, M. Zhang, and S.-Q. Yang, "Deploying P2P networks for large-scale live video-streaming service," IEEE Commun. Mag., vol.45, no.6, pp.100–106, 2007.

[10] F. Wang, Y. Xiong, and J. Liu, "mTreebone: A collaborative tree-mesh overlay network for multicast video streaming," IEEE Trans. Parallel Distrib. Syst., vol.21, no.3, pp.379–392, 2010.

[11] T.T. Do, K.A. Hua, and M.A. Tantaoui, "P2VoD: Providing fault tolerant video-on-demand streaming in peer-to-peer environment," Proc. IEEE Conf. on Communications, pp.1467–1472, 2004.