A Cost-Effective Buffer Map Notification Scheme for P2P VoDs Supporting VCR Operations

Ryusuke UEDERA∗, Student Member and Satoshi FUJITA†a), Member

SUMMARY In this paper, we propose a new buffer map notification scheme for Peer-to-Peer Video-on-Demand systems (P2P VoDs) which support VCR operations such as fast-forward, fast-backward, and seek. To enhance the fluidity of such VCR operations, we need to refine the size of each piece as small as possible. However, such a refinement significantly degrades the performance of buffer map notification schemes with respect to the overhead, piece availability and the efficiency of resource utilizations. The basic idea behind our proposed scheme is to use a piece-based buffer map with a segment-based buffer map in a complementary manner. The result of simulations indicates that the proposed scheme certainly increases the accuracy of the information on the piece availability in the neighbor-
hood with a sufficiently low cost, which reduces the intermittent waiting time of each peer by more than 40% even under a situation in which 50% of peers conduct the fast-forward operation over a range of 30% of the entire video.

key words: Peer-to-Peer, video-on-demand, buffer map, VCR operation

1. Introduction

Video-on-Demand (VoD) is an online service which enables users to watch their favorite videos at any time, at any place [1], [2]. In conventional VoD systems, each client consecutively acquires a requested video file from a server in such a way that she can start watching while the download is still in progress. However, as is pointed out by many researchers [11], [12], such a centralized approach causes several critical issues such as the access bottleneck and a single point of failure at the central server. In order to overcome such issues, Peer-to-Peer (P2P) technology has recently emerged as a perspective way to realize scalable, dependable VoDs.

A VoD system based on the P2P technology is generally referred to as P2P VoD in the literature. P2P VoD is a distributed system consisting of many autonomous computers called peers, and each peer participating in the system plays the roles of a server and a client at the same time, so that the content delivery is realized with the aid of many participating peers. Such a cooperative behavior of the participants reduces the load of the central server [11], [17], and in many cases, it significantly improves the scalability and the dependability of the overall system.

The usability of VoD systems could be evaluated by the quality of functions supported by the system in addition to the quantitative metrics such as the resolution, response time, and the intermittent waiting time. VCR (Video Cassette Recorder) function such as seek and fast-forward is a representative of such high quality functions. In fact, there are many commercial VoD systems supporting VCR function, such as DVD Empire [3], Charter On Demand [4], and MCV Video On Demand [5]. Even for P2P VoDs, the sup-port of VCR function is increasingly becoming a key issue to realize highly available VoD systems [8], [19].

In general P2P VoDs, each video file is divided into small chunks called pieces, and is exchanged among neighboring peers in the overlay network (i.e., P2P overlay) so that the set of pieces held by each peer monotonically increases. With this notion, seek is defined to be an operation which moves the playing position to a piece designated by the user and fast-forward is an operation which selectively plays one arbitrary piece out of α consecutive pieces for a given α (parameter α is an integer indicating the speed of the fast-forward operation, which is assumed to be given by the user beforehand). The configuration of the set of acquired pieces, which indicates the piece availability in the corresponding peer, is represented by using the notion of buffer maps, which is generally realized in either piece-based or segment-based manner (see Sect. 2 for the details). During the download of a file, peers exchange their buffer maps among neighbors, and make a decision on the selection of pieces to be downloaded by the received buffer maps [18].

The support of VCR function introduces several issues to the design of P2P VoDs. For example, to enhance the fluidity of fast-forward (and fast-backward) operations, the size of each piece must be much smaller than conventional P2P VoDs, and the support of the seek operation, which allows users to jump to arbitrary chapters in the video file, violates the assumption behind conventional P2P VoDs such that each peer tries to collect as many pieces close to the current playback position as possible [14], [16]. In each of the above two cases, precise identification of the buffer map of the neighbors would play a crucial role in efficiently supporting VCR function. In fact, smaller pieces will shorten the time which can be spent for each piece and it increases the number of candidate pieces to be acquired from the neighbors. If we could not assume the continuity of pieces
to be downloaded, we need to calculate pieces to be acquired from the neighbors by referring to the piece availability in its neighborhood as well as the priority of pieces determined by an appropriate piece selection policy. However, as the number of pieces corresponding to the given file increases, conventional piece-based or segment-based buffer map schemes will cause a significant overhead or a low accuracy of the notified information (see Sect. 2 for the details).

In this paper, we propose a buffer map notification scheme for P2P VoDs supporting VCR operations which complementarily uses piece-based and segment-based buffer maps. By taking such an approach, peers can acquire an accurate piece availability in their neighbors without significantly increasing the overhead. In addition to the notification scheme, we propose a piece selection policy which takes advantage of the proposed notification scheme. The proposed policy enables peers to exchange pieces more flexibly and more efficiently than conventional policies. The performance of the proposed scheme is evaluated by simulation. The result of simulations indicates that the proposed scheme reduces the intermittent waiting time of each peer by more than 40% when 50% of peers conduct the fast-forward operation over a range of 30% of the entire video.

The remainder of this paper is organized as follows. Section 2 describes related works including existing buffer maps. Section 3 describes the proposed scheme. Section 4 summarizes the result of performance evaluation, and finally, Sect. 5 concludes the paper with future works.

2. Related Work

Recently, many works have tried to support VCR operations in P2P VoDs. Most of such works focus on the way of constructing index overlays which efficiently support VCR operations [7], [8], [10], [19].

In RINDY overlay management scheme [8] adopted in GridCast [9], each peer maintains a set of concentric rings with power law radii and places its neighbors on these rings according to their playback positions. More concretely, neighbors which have a playback position close to the center peer are placed on the innermost ring to exchange their video data with the center peer, and the remaining neighbors are placed on the outer rings to be used as the routers to the peers to have a playing position specified by the center peer when it invokes a seek. Temporal-DHT [7] is an index overlay based on the notion of distributed hash table (DHT). In P2P VoDs based on Temporal-DHT, each peer caches video data around its current playing position and registers its buffer to the DHT. Temporal-DHT exploits lazy updates of the DHT records and provides estimated results for given queries in order to deal with the dynamics of cached contents. Dynamic Skip List (DSL) [19] is a randomized data structure consisting of a set of layers. In DSL, each peer uses its playback position as the key. Each peer arriving at the system initially joins the base layer and it randomly and independently promotes itself to the upper layers. Each peer has logical links to its neighbors in each layer, which can be used to find a data provider to have a required key while skipping unnecessary peers. In [10], the authors proposed Buffer Assisted Search (BAS) structure, which can reduce the size of existing index overlays such as DSL [19] by removing redundant peers. More concretely, BAS structure removes peers whose buffers are fully covered by other participants from the index overlay, which reduces the control overhead and achieves the fast search of peers.

Existing buffer map notification schemes are based on the notion of piece or segment. A piece-based buffer map, which is referred to as \textbf{Piece-BM}, is a bit-array of the same length with the total number of pieces in the corresponding file [11], [18], where an element in the array takes value 1 if and only if the corresponding piece is held by the peer (note that a buffer map always starts with an array with all 0’s and ends at an array with all 1’s). The fact that a peer acquires a piece is recorded to its local array, and the peer \textit{immediately} notifies the fact to all neighbors. Thus, each peer can accurately learn the piece availability in its neighborhood by (locally) referring to the corresponding buffer maps. However, it causes the waste of resources such as memory, CPU and bandwidth due to the large size of buffer maps and the frequent exchange of buffer maps.

A segment-based buffer map, which is referred to as \textbf{Segment-BM}, was proposed to overcome such a large overhead of Piece-BM. In Segment-BM, a segment of several pieces is used as the basic element in the buffer map instead of simple pieces. Although each peer collects pieces from neighbors as in Piece-BM, it does not immediately notify the acquisition of each piece. Instead, it notifies the status when it encounters the collection of all pieces in a segment. Such an approach would certainly overcome the large overhead of Piece-BM. However, it is not enough to support VCR operations, since a simple application of Segment-BM to VCR operations such as fast-forward and seek operations will increase the number of “incomplete” segments, which will not be notified to the neighbors until they become a “complete” segment.

3. Proposed Scheme

In this section, we propose a complementary piece-based buffer map (\textbf{Comp-BM}, for short) to overcome the inefficiency of Segment-BM during the execution of VCR operations. Let us assume that the given video file is divided into several pieces with identifier starting from 1, where the first piece in the first segment is given identifier 1. The role of Comp-BM is to carry the information on acquired pieces which are not notified to the neighbors under Segment-BM. As will be described later, in the proposed scheme, Comp-BM is used as a complementary of Segment-BM, in the sense that:

1. the piece availability in a complete segment is notified to the neighbors using Segment-BM and
2. the piece availability in an incomplete segment is notified using Comp-BM.
Each Comp-BM consists of several fragments, where each fragment corresponds to a segment and is represented by a pair of the identifier of the segment and a bit array representing the piece availability in the segment. For example, Comp-BM (1:1100, 3:0011) indicates two incomplete segments consisting of four pieces each. Then, if a peer receives a Comp-BM (1:1100, 3:0011) from a neighbor, the receiver can learn that the sender holds pieces 1, 2, 11 and 12 without receiving the whole bit array as in Piece-BM (recall that in Segment-BM, such an incomplete segments are never notified to the neighbors). The reader should note that each Comp-BM contains only bit arrays of incomplete segments; i.e., by allowing users to designate identifiers of segments, it becomes more flexible than Segment-BM and becomes more efficient than Piece-BM.

3.1 Basic Procedure

There are several issues to be considered in designing P2P VoD with the notion of Comp-BM. The first issue is how to determine the size of Comp-BM to be notified to the neighbors and the second issue is how to determine the timing of notifications to the neighbors (the other issues will be considered in later subsections). The reader should note that such issues do not occur in Segment-BM. In fact, the size of Segment-BM is fixed to the number of segments and notification is issued by each peer when it collects all pieces in a segment. The reader should note that those two issues significantly affect the traffic load over the network and the accuracy of piece availability in the neighborhood.

In our proposed scheme, each peer issues a Comp-BM if one of the following three conditions is satisfied:

Case 1: When it finishes a VCR operation.
Case 2: Let \( p \) be the playback position at the time of completing a VCR operation and \( q \) be the end piece of the segment containing \( p \). If the segment containing \( p \) is incomplete at the time of acquiring piece \( q \), it issues a Comp-BM corresponding to the segment when it acquires the end piece \( q \).
Case 3: When it connects to a new peer.

In each case, the scheme firstly determines a “potential range” concerned with the operation which consists of consecutive segments, and then identifies segments to be included in the Comp-BM by excluding complete and untouched segments from it, where a segment is said to be untouched if it contains no acquired pieces. The potential range in the first case is determined as follows. If the executed VCR operation is either fast-forward or fast-backward, then all (consecutive) segments covered by the operation are included in the potential range. For example, if a peer conducts a fast-forward from piece 3 to piece 9 and acquired pieces 4, 6, 7 and 9 in this order, since pieces 3 and 9 are contained in segments 1 and 3, respectively, the potential range of segments is determined as [1, 3]. Then it excludes complete segment 1, and issues a Comp-BM (2:0110, 3:1000). If the execution time of the fast-forward or fast-backward operation is too long (e.g., few minutes), it is possible to divide the range into several sub-ranges so that the information on each sub-range is immediately notified as an interim report using several (consecutively issued) Comp-BMs. If the operation is seek, on the other hand, only the segment being downloaded at the time of invocation is included in the potential range. For example, if it invokes a seek to piece 10 while downloading piece 6 under a normal playback mode, the potential range is determined as [2, 2]. Then it issues a Comp-BM (2:1100) since segment 2 is neither completed nor untouched. Note that if the peer continues another normal playback starting from piece 10 and acquires the end piece 12 of the segment containing piece 10, it generates another Comp-BM including the segment, and notifies it to the neighbors.

In the second case, the size of the potential range is determined in the same way for the first case, i.e., only the segment containing the new playback position after finishing a VCR operation is included in the potential range. For example, if the playback position after finishing the seek is piece 10, and it downloads pieces from 10 to 12 without downloading piece 9. In such a case, it determines the potential range as [3, 3], and issues a Comp-BM (3:0111).

Finally, in the third case, the potential range equals to the whole of the segments, i.e., it informs all segments except complete and untouched segments to the newly connected peer as a Comp-BM (note that it also informs its entire Segment-BM to the peer).

3.2 Management of Comp-BMs

In the proposed scheme, each peer maintains the information contained in Comp-BMs received from neighbors as well as Comp-BMs issued by itself. Those fragments of information will be referred by each peer, being combined with Segment-BMs received so far, in order to accurately recognize the piece availability in its neighborhood. A key challenge in this aspect is how to efficiently manage such information with as small overhead as possible, even if the number of pieces grows and each user frequently invokes VCR operations. More concretely, the set of Comp-BMs received from a neighbor should be maintained so that it can return the piece availability in a designated incomplete segment as quickly as possible.

To this end, in the proposed scheme, we maintain such fragments of information in the form of a collection of Comp-BMs to have a range which coincides with the range of a segment. Let \( m \) be the number of segments associated with a video file. The maintenance scheme starts with \( m \) bit arrays of all 0’s, each of which represents to the piece availability in the corresponding segment. The reader should note that the size of each of \( m \) bit arrays equals to the size of each segment. After receiving a Comp-BM from a neighbor, it copies each bit array in the Comp-BM to its corresponding bit array. If a segment becomes complete after the write, it removes the corresponding Comp-BM from the set of Comp-BMs since the piece availability in complete
segments can be referred through Segment-BM in piece selection schemes.

3.3 Proposed Piece Selection Policy

As was explained above, the use of Comp-BMs increases the accuracy of information on the buffer map of the neighboring peers and enables participant peers to acquire more pieces from neighbors compared with conventional segment-based schemes, which would result in a more flexible, more efficient piece selection policy. In this subsection, we propose a new piece selection policy based on the notion of Comp-BMs.

In the proposed policy, each peer tries to sequentially acquire available pieces in a segment from a neighbor (i.e., it adopts In-Order policy described in [15]). If there is a neighbor which holds a complete copy of a segment, which is easily recognized by referring to the collection of complete segments, it can acquire all pieces in the segment from the neighbor to reduce the communication overhead. Even if there is no such peer in its neighborhood, if there exists a neighbor which notified a Comp-BM intersecting with the segment, it can download pieces in the segment from the neighbor, although it would need additional time to complete the acquisition of the whole segment. The reader should note that although we could refine the above piece selection scheme by using the notion of randomization as in [18], we leave such a refinement as a future work.

The information contained in Comp-BMs would also be useful to realize a load balancing among neighbors. For example, if there are several neighbors holding a partial copy of a segment, it can periodically change the target of download (by using the round-robin rule, for example) so that the load of uploaders could be evenly balanced. In fact, if the ratio of incomplete segments is sufficiently large compared with complete segments notified by conventional segment-based schemes, the use of Comp-BMs significantly increases the chance of uploading pieces to the neighbors. Again, although the performance of a simple round-robin rule could be refined by directly referring to the load of each peer (as in many load balancing schemes) and by adopting the Tit-for-Tat strategy (as in BitTorrent), we leave such a refinement as a future work.

4. Evaluation

4.1 Setup

We conducted several simulations to evaluate the performance of the proposed scheme. In simulations, we assume the existence of a media server which stores all video files in its local storage. Each peer can access the media server not only as the first provider of the video data, but also as the last resort when it fails to acquire a piece from its neighbors, where an intermittent suspension of the video play occurs at a peer when it could not acquire a piece at the time of playing the piece. Note that such a suspension could occur even in the environment with a media server, since the upload bandwidth of the media server is limited; i.e., a concentration of many upload requests will significantly degrade the upload performance of the media server.

4.1.1 Scenario

As for the behavior of the peers, we will consider the following scenario: Each peer arrives at the system according to the Poisson distribution with average interval of two seconds, where initially, every newly arrived peer has no pieces. Upon arrival, it downloads pieces corresponding to the first two seconds from the media server. Then, after completing such download, the playback of the video and the piece exchange with the other peers immediately start.

Each peer can conduct either seek or fast-forward during the playback of the video, where seek is an operation which moves the playing position to a piece designated by the user (i.e., any piece can be designated as the destination of seek operation) and fast-forward is an operation which selectively plays one arbitrary piece out of α consecutive pieces for a given α. Parameter α is an integer indicating the speed of the fast-forward operation, which is assumed to be given by the user beforehand. For example, when α = 3, it selects one piece from pieces 1, 2, and 3; one piece from pieces 4, 5, and 6; and so on. As for the execution of such VCR operations we consider the following two scenarios:

Scenario A: 50% of peers conduct fast-forward operation in a double speed (i.e., α = 2) over a range of r % of the entire video, where the start position of the fast-forward is determined according to the uniform distribution, while the remaining half do not conduct any VCR operations. During the execution of a fast-forward, each peer sends an interim report to its neighbors every five seconds. The result under this scenario is described in Sect. 4.3.

Scenario B: 50% of peers conduct x forward-seek operations over a distance of 5 % of the entire video, which corresponds to 75 seconds, where the start position is determined according to the uniform distribution, while the remaining half do not conduct any VCR operations. The result under this scenario is described in Sect. 4.4.

Note that those scenarios reflect observations on existing P2P VoD systems [8], [9] in the sense that many users conduct VCR operations and the distance of each VCR operation is not very large, although it is simplified compared with actual behavior of the users. In each scenario, the playback of a peer terminates if it reaches the end of the file, and a peer leaves the system as soon as it terminates the playback.

4.1.2 Metrics

The performance of the schemes is evaluated with respect to the following four metrics:

1. The number of un-notified pieces Np which are acquired from a neighbor but are not notified to the other
Table 1  Average memory usage under fast-forward operation.

| Method      | Segment-BM | Proposed |
|-------------|------------|----------|
| $U_m$ [KB]  | 4.496      | 45.845   |

neighbors. Note that this metric represents the accuracy of the scheme.

2. The traffic per peer caused by the exchange of buffer maps. We will assume TCP as the transport layer protocol, and represent such traffic by $T$ [Kbps].

3. The average wait time $W$ [s] of a peer due to the (intermittent) suspension of the video play.

4. The size of memory $U_m$ [KB] which is necessary to keep the buffer map of the neighbors including itself.

Each result is an average over 20 runs unless otherwise stated. To exclude possible ambiguity, we call a method using Segment-BM the “Segment-BM method,” and a method using Comp-BM the “proposed method.”

In the following, we will merely compare the performance of the proposed scheme with Segment-BM, since the traffic per peer of Piece-BM is much higher than Segment-BM, e.g., if each segment consists of 32 pieces, the traffic per peer of Piece-BM becomes 32 times of Segment-BM. The memory usage of Piece-BM is almost the same with the proposed scheme. As for the other metrics, we could not observe a significant difference between Piece-BM and the other schemes, although the wait time and the number of un-notified pieces certainly become smaller than the proposed scheme as the precision of learned buffer map increases by using Piece-BM.

4.1.3 Parameters

The other parameters used in the simulation are as follows (those values are determined by referring to previous works such as [6], [13]). In each run of the simulation, the total number of peers arriving at the system is 300. All peers are homogeneous, and each peer can maintain at most 30 connections to the other peers. Communication bandwidth of each peer is 1024 [Kbps] in each of upload/download directions, and the communication bandwidth of the media server is 50 [Mbps]. We assume that the system contains exactly one video file of length 1500 [s], with the playback bit-rate of 512 [Kbps]. The size of each piece is 64 [Kb], and the size of each segment is 2 [Mb].

4.2 Memory Usage

At first, we compare the average memory usage of two methods. Note that this metric is independent of the type of scenario, since it is merely dependent on the type of method, the size of the neighborhood, and the granularity of video segmentation. Table 1 summarizes the result. The memory usage in the proposed method is about ten times of the Segment-BM method, since it should maintain bit arrays associated with all segments for each neighbor. However, even in the proposed method, the actual value of the memory usage is small enough, e.g., less than 50 KB for video of 25 min, to be realized in modern personal computers.

4.3 Performance under Fast-forward Operation

Next, we compare the performance of two methods under Scenario A. Figure 1 shows the number of un-notified pieces $N_p$, where the percentage $r$ of the video scanned by the fast-forward operation is fixed to 25%. Each point in the figure is the result of one simulation and the horizontal axis is the elapsed time. We can observe that in the Segment-BM method, the value of $N_p$ progressively increases as the elapsed time increases, and it reaches nearly 155,000 at the peak time. In the proposed method, on the other hand, the number is bounded by 6,500 during the simulation time, which implies that the exchange of Comp-BMs significantly enhances the piece availability in the P2P VoD. The reader should note that the proposed method can not reduce $N_p$ to 0, since the piece availability in a segment which is currently downloaded under the normal playback, will not be notified until completing the download.

Figure 2 summarizes the results on the other two metrics, where in those results, we vary parameter $r$ from 0% to 50% (note that $r = 0$ corresponds to the situation in which no peer conducts VCR operations). The result on the average traffic per peer is shown in Fig. 2(a). The traffic in the proposed method increases as $r$ increases, which is because the amount of Comp-BMs exchanged among peers also increases, although the traffic in the Segment-BM method fluctuates only slightly. The superiority of the proposed method could also be demonstrated by the average wait time $W$. Figure 2 (b) shows the result. We can see from the figure that in both methods, $W$ decreases as $r$ increases for small $r$’s, while for large $r$’s exceeding 20%, $W$ increases as $r$ increases. Such an interesting phenomenon is caused by the change of the number of played pieces and the number of shared pieces. More precisely, for small $r$’s, the number of pieces “skipped” by the peers increases as the range of the fast-forward increases, so that peers are exempted to collect bottleneck pieces. However, for large $r$’s, the number of shared pieces also decreases due to the skip of the pieces, which increases the number of bottleneck pieces. The fig-
(a) Average traffic per peer.

(b) Average wait time of peers.

Fig. 2 Performance under Scenario A for different $r$'s.

Figure also shows that the increase of $W$ in the Segment-BM is more significant than the increase in the proposed method, which is due to the increase of the number of un-notified pieces. For example, the proposed method reduces $W$ by more than 40% of the Segment-BM method when $r \geq 30$. Note that the traffic caused by the exchange of Comp-BMs has a little impact on $W$, since it accounts for only a small fraction of the total traffic in data exchanges.

### 4.4 Performance under Seek Operation

Finally, we compare the performance of the two methods under Scenario B. Figure 3 shows the number of un-notified pieces $N_p$ where parameter $x$, which represents the number of invocations of the seek operation per peer, is fixed to five. Each value in the figure is the result of one simulation and the horizontal axis is the elapsed time. Similar to the case of fast-forward, the proposed method bounds the value of $N_p$ by a small constant, although that of the Segment-BM method increases as the elapsed time increases. However, the maximal value of $N_p$ in the Segment-BM method is not large (e.g., it is only about 18,000), since each seek operation causes un-notified pieces only within the segment being downloaded.

Figure 4 summarizes the results on the other metrics where $x$ is varied from 0 to 10 (note again that $x = 0$ corresponds to the situation in which no peer conducts VCR operations). As shown in Fig. 4 (a), similar to the case of fast-forward, the traffic in the proposed method gradually increases as $x$ increases, since it increases the amount of Comp-BMs exchanged among peers. The result on the average wait time $W$ is shown in Fig. 4 (b). Unlike the case of the fast-forward, in both methods, $W$ does not increase even for large $x$'s, which is due to the difference of the way of piece skipping. More precisely, unlike the fast-forward which skips only several pieces within a range, the seek operation skips all pieces between the start and the destination pieces, which is less sensitive to the existence of bottleneck pieces. We can also observe that with respect to the wait time $W$, there is no difference between two methods. It is probably due to the fact that the number of un-notified pieces is small, i.e., such a small difference of the observed piece availability could be compensated by conducting a download from other neighbors. Another interesting observation we could make from the figure is that $W$ of both methods significantly decreases by increasing $x$ from 0 to 1, i.e., it decreases from about 1600 seconds to 150 seconds. It is probably due to
the difference of playing positions. More precisely, if several peers conduct seek operation, playing positions of those peers will be spread over the video file from the beginning to the end of the file, which causes an active exchange of pieces among peers. On the other hand, if no peer conducts seek operation, playing positions of those peers will be concentrated to some point in the video file, which increases the number of bottleneck pieces since many peers tend to request similar pieces around the concentrated point.

By the above observations, we can conclude that the impact of the proposed method is not very large for the seek operation, although as was shown in the last subsection, it is significant for the fast-forward operation.

5. Concluding Remarks

In this paper, we proposed a buffer map notification scheme for P2P VoDs which support VCR operations. In conventional segment-based buffer maps, several pieces acquired from a neighbor can not be notified to the other neighbors, which significantly degrades the piece availability in the neighborhood and disturbs an efficient exchange of the pieces among them. We overcome problems in conventional schemes by combining piece-based buffer maps with a segment-based buffer map. The result of extensive simulations indicates that our scheme can enhance the piece availability in the neighborhood to reduce the intermittent wait time of each peer by 40% when 50% of peers conduct fast-forward operations. As a future work, we need to develop a prototype system to evaluate the influence of the waste of resources in detail. It is also interesting to investigate how small size of pieces is enough to make the fast-forward operation to be fluid.

Acknowledgements

This work was supported in part by the Scientific Grant-in-Aid from Ministry of Education, Science, Sports and Culture of Japan and the Telecommunications Advancement Foundation.

References

[1] YouTube. http://www.youtube.com/
[2] Hulu. http://www.hulu.com/
[3] DVD Empire. http://www.dvdempire.com/
[4] Charter. http://www.charter.com/
[5] MCV. http://www.mcvguam.com/
[6] A.R. Bharambe, C. Herley, and V.N. Padmanabhan, “Analyzing and improving a BitTorrent networks performance mechanisms,” Proc. INFOCOM, pp.1–12, 2006.
[7] A. Bhattacharya, Z. Yang, and S. Zhang, “Temporal-DHT and its application in P2P-VoD systems,” Proc. 2010 IEEE International Symposium on Multimedia, pp.81–88, 2010.
[8] B. Cheng, H. Jin, and X. Liao, “Supporting VCR functions in P2P VoD services using ring-assisted overlays,” Proc. IEEE International Conference on Communications, pp.1698–1703, 2007.
[9] B. Cheng, L. Stein, H. Jin, X. Liao, and Z. Zhang, “GridCast: Improving peer sharing for P2P VoD,” ACM Trans. Multimedia Computing, Communications, and Applications (TOMCCAP), vol.4, no.4, pp.1698–1703, 2007.
[10] H. Chi, Q. Zhang, J. Jia, and X. Shen, “Efficient search and scheduling in P2P-based media-on-demand streaming service,” IEEE J. Sel. Areas Commun., vol.25, no.1, pp.119–130, Jan. 2007.
[11] C. Dana, D. Li, D. Harrison, and C.-N. Chua, “BASS: BitTorrent assisted streaming system for video-on-demand,” Proc. International Workshop on Multimedia Signal Processing, pp.1–4, 2005.
[12] Y. Guo, K. Suh, J. Kurose, and D. Towsley, “P2Cast: Peer-to-peer patching scheme for VoD service,” Proc. 12th International Conference on World Wide Web, pp.301–309, 2003.
[13] Y. Huang, T.-Z.-J. Fu, D.-M. Chiu, J.-C.-S. Lui, and C. Huang, “Challenges, design and analysis of a large-scale P2P-VoD system,” Proc. ACM SIGCOMM 2008 Conference on Data communication, pp.375–388, 2008.
[14] B. Liu, Y. Cui, B. Chang, B. Gotow, and Y. Xue, “BitTube: Case study of a Web-based peer-assisted video-on-demand system,” Proc. Tenth IEEE International Symposium on Multimedia, pp.242–249, 2008.
[15] N. Parvez, C. Williamson, A. Mahanti, and N. Carlsson, “Analysis of BitTorrent-like protocols for on-demand stored media streaming,” SIGMETRICS Perform. Eval. Rev., vol.36, no.1, pp.301–312, 2008.
[16] S. Sakashita, T. Yoshihisa, T. Hara, and S. Nishio, “A data reception method to reduce interruption time in P2P streaming environments,” Proc. 13th International Conference on Network-Based Information Systems, pp.166–172, 2010.
[17] R. Uedera and S. Fujita, “Adaptive prefetching scheme for Peer-to-Peer video-on-demand systems with a media server,” IEICE Trans. Inf. & Syst., vol.E94-D, no.12, pp.2362–2369, Dec. 2011.
[18] A. Vlavianos, M. Iliofotou, and M. Faloutsos, “BiToS: Enhancing BitTorrent for supporting streaming applications,” Proc. 25th IEEE International Conference on Computer Communications, pp.1–6, 2006.
[19] D. Wang and J. Liu, “A dynamic skip list-based overlay for on-demand media streaming with VCR interactions,” IEEE Trans. Parallel Distrib. Syst., vol.19, no.5, pp.503–514, 2008.