ISSN 2047-3338

Buffer Efficient Fast Broadcasting Scheme

Mohammad Saidur Rahman, Mohammad Saiedur Rahaman,
Ahmed Ridwanul Islam and Abhijit Bhowmik

***Abstract*— A popular video can be broadcasted by partitioning the video into segments, which are broadcast on several channels simultaneously and periodically. This approach permits multiple users to share channels that lead to higher bandwidth employment. Previous schemes generally focus on reducing clients' waiting time. This work studies another important issue, namely client buffer1 savings. A reverse fast broadcasting (RFB) scheme is introduced to improve the buffer problem. RFB has the identical waiting time like FB. We introduced a new Buffer Efficient Fast Broadcasting scheme that has smaller buffer requirement and less waiting time than RFB and FB.**

***Index Terms*— VOD, Buffer Efficient, Fast Broadcasting and Near VoD Scheme**

I. INTRODUCTION

VIDEO on demand (VOD) systems have become achievable due to progress in broadband networking technology and the enlargement of processor speed and disk capacity [1], [2]. A VOD system is typically implemented by client-server architecture, and may easily run out of bandwidth, since the growth in bandwidth can never keep up with the growth in the number of clients. To improve the problem, many results, known as near-VOD services, have been introduced by sacrificing some VCR functions, such as fast forward or rewind. A solution is to just broadcast popular videos. According to [3], a few (10 or 20) very popular videos constitute 80% of viewers' requests. Because the server's broadcasting activity is independent of the arrivals of requests, this approach is appropriate to popular videos that may interest many viewers at a particular period of time. An approach to broadcast a popular video is to divide it into segments. A video server then concurrently transmits the segments on different data channels. One channel transmits the first segment in real time, while the other channels send the remaining segments according to a schedule predefined by the various schemes. When clients want to watch a video, they wait for the start of the first segment on the first channel. Therefore, their greatest waiting time equals the length of the first segment. While the viewers are watching the video, their set-top boxes (STB) or computers have to download and buffer enough data from the other channels to enable them playing the video segments in turn.

The previous studies [4]–[15] mainly focus on reducing client waiting time. The projected schemes typically involve their clients to buffer about 40–50% of a playing video. Assume that a video of 120 minutes is played at a rate of 1.5 Mbps. The video size equals 1350 Mbytes. The buffered videos are 540–635 Mbytes, and thus an STB must be equipped with either hard disks or RAM to store the video data. These additional storage devices increase the cost of STB, and make an entire video system costly. In Paper [17], Reverse First Broadcasting (RFB) scheme tries to reduce this buffer requirement by reversing the segment allocation policy as specified in Fast Broadcasting (FB) scheme [6]. This scheme reduces the buffer requirements significantly at client side. In our work, we introduce a new scheme, Buffer Efficient Fast Broadcasting (BEFB) scheme, based on the FB scheme in [6] and RFB scheme in [17] that reduces the buffer requirement more than RFB. This work also reduces the user waiting time by increasing the number of segments than FB and RFB.

The remainder of this paper is organized as follows. Section 2 reviews the related work. The BEFB scheme is introduced in section 3. Section 4 provides the analysis and comparison of BEFB with FB and RFB schemes. Results from this work are discussed in section 5. Section 6 concludes this paper.

II. RELATED WORKS

Reducing waiting time is one of principle goals for the VOD researchers. To achieve this goal several schemes have been introduced by time. The staggered broadcasting [4] scheme requires a server to allocates k channels to send a video. This method treats a complete video as a single segment, and broadcasts it on each channel at different start times. The maximum waiting time for viewers is L/k, where L represents the video length. The PB scheme [5] partitions a video into k segments of geometrically increasing sizes. The

Manuscript received July 11, 2011.

Mohammad Saidur Rahman is with the American International University-Bangladesh, Banani, Dhaka, Bangladesh (e-mail: saidur@ aiub.edu).
Mohammad Saiedur Rahaman is with the American International University-Bangladesh, Banani, Dhaka, Bangladesh (e-mail: saied@ aiub.edu).
Ahmed Ridwanul Islam is with the American International University-Bangladesh, Banani, Dhaka, Bangladesh (e-mail: aridwan@ aiub.edu).
Abhijit Bhowmik is with the American International University-Bangladesh, Banani, Dhaka, Bangladesh (e-mail: abhijit@ aiub.edu).

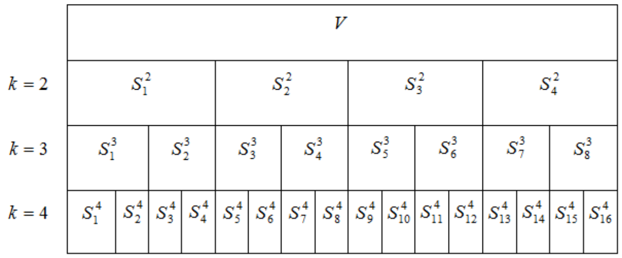

Fig. 1. The relationship of segments when k (= 2, 3, 4) channels are assigned to a video V in Buffer Efficient Fast Broadcasting (BEFB) scheme

geometric series has a factor α , where $\alpha > 1$. Given a fixed bandwidth, the method outperforms the staggered broadcasting scheme on the waiting time. Furthermore, the FB scheme [6], [7] obtains shorter waiting time than the staggered broadcasting and PB schemes. FB is also simple to implement on IP networks [16].

The theoretical boundaries of waiting time and bandwidth consumption have also been investigated. The harmonic broadcasting (HB) scheme [11] initially divides a video into several segments equally, and further divides the segments into sub-segments according to the harmonic series. Yang, Juhn and Tseng [12] proved that HB yields the minimum waiting time at a fixed bandwidth. The cautious harmonic broadcasting (CHB) [13], quasi-harmonic broadcasting (QHB) [13] and polyharmonic broadcasting (PHB) schemes [14] were proposed to keep on-time data delivery. Harmonic-series schemes have smaller waiting time than the non-harmonic schemes, but are more complicated and less practical owing to the multitude of allocated channels [8], [15]. For example, given a video of 100 minutes and a waiting time of 10 seconds, CHB needs a video server to broadcast data on 600 channels simultaneously, while FB simply allocates 10 channels. This study thus focused on client buffer savings in non-harmonic schemes.

A. Review of FB Scheme

In FB scheme, a video V of length D is divided into $2^k - 1$ segments, where k is the number of channels allocated to V . The segments can be expressed as $S_{2^i}^k \bullet S_{2^i+1}^k \bullet ..... \bullet S_{2^k-1}^k$ .The server then periodically broadcasts segments S_{2^i} to S_{2^i-1} on channel C_i , where, $i = 0,1,...,k-1$. When a client wants to watch a video, the client waits for the beginning of the first segment on the first channel, and downloads all the un-received segments on each channel. Thus, the maximum waiting time equals the length of the first segment (i.e. $D/2^k - 1$). The maximum number of segments buffered by a client is $2^k - 1$. In other words, a client must buffer a half of a playing video in the worst case (i.e. $\left(2^{k-1}/2^k - 1\right) \approx 58\%$).

B. Review of RFB Scheme

The FB scheme mainly concentrates on decreasing waiting time. It is not buffer efficient. Reversed Fast Broadcasting Scheme [17] works on reducing the buffer requirement at the client side. The segmentation and allocation of segments of RFB on channel C_i is similar to FB. But the major difference between FB and RFB is that RFB broadcasts the segments in reverse order. When the video segments are available to download in a time slot, it first checks whether the segment will be available again before it requires playing at the client side or not. If the segment be available again before it plays, it does not download the segment now. It will just skip downloading the segment. Otherwise, it will download the segment. In this way, RFB reduces the buffer requirement to about 38%.

III. Proposed Scheme

In this section, our proposed scheme, Buffer is discussed in details. The proposed scheme, Buffer Efficient Fast Broadcasting scheme (BEFB), is a proactive approach to broadcast a video. The video is divided into segments and repeatedly broadcasted on some channels. The client downloads the segments available in a particular time slot maintaining some rules. The objective of this scheme is to reduce the buffer requirements at the client side. The details of this scheme are discussed below.

If the server allocates k channels to a video, the channels are named as C_i^k where k is the allocated channel number. i is the channel index. The value of the channel index i starts from 0 to k-1, i.e. $i = 0,1,...,k-1$. The video segments are grouped sand transmitted repeatedly through a channel (see Fig- 2). The groups are named as G_i^k , where k is the allocated channel number and i is the channel index through which the group will be broadcasted repeatedly. If, $i < 2$, the contents of the group G_i^k are the segments $S_{2^i}^k \bullet S_{2^i+1}^k \bullet ..... \bullet S_{2^{i+1}-1}^k$ and if $i \geq 2$, G_i^k contains the segments $S_{2^i}^k \bullet S_{2^i+1}^k \bullet ..... \bullet S_{2^{i+1}}^k$.

Now, consider a client wants to watch a video which is being transmitted by the server maintaining the rules specified above. Let, the time when the server started the transmission of the video is V_{start} and V_{play} is the time when the client gave the PLAY command.

Here, V_{start} must be lesser than V_{play} . V_{play} is the origin of the time axis throughout the paper. Client cannot start watching the video immediately as the first segment must be downloaded first. Hence, the client has to wait for at least one time unit. The client plays the video in order of at $V_{play} + 1$. During the first time unit client end device may download at best k segment(s) available at all k channels. Let, T_{Now} is the time when the client device wants to download a video segment x and T_{Next} is the time when the segment x is available again to download at channel C_i^k . If $i < 2$, then $T_{Next} = T_{Now} + 2^i$ and if $i \geq 2$, then $T_{Next} = T_{Now} + 2^i + 1$. The client is assumed to play the segment S_x at time T_{Use} . The T_{Now} , T_{Next} and T_{Use} leading to an inequality

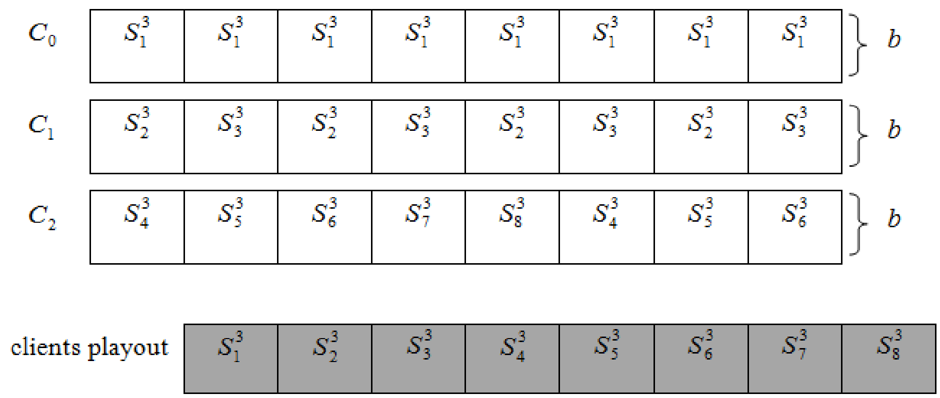

Fig. 2. The channel allocation for Buffer Efficient Fast Broadcasting (BEFB) Scheme with k=3

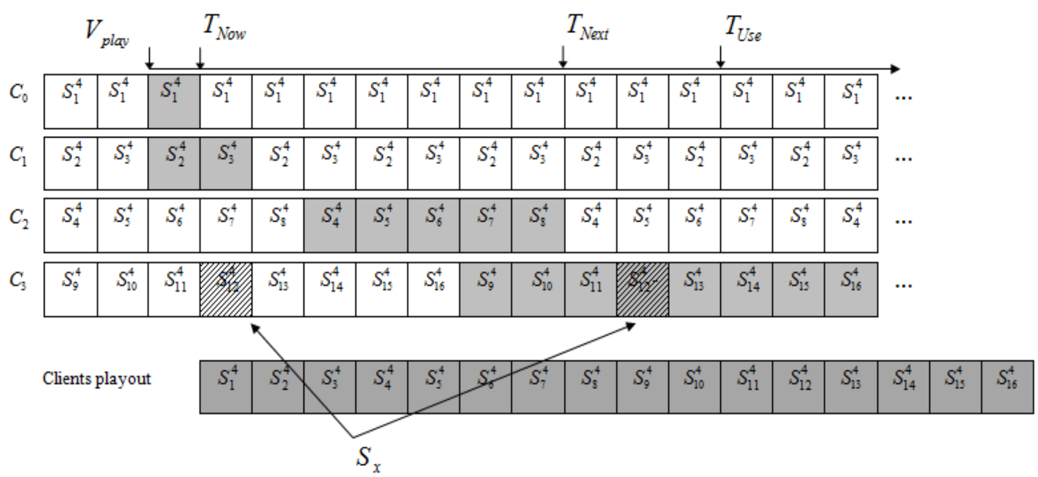

Fig. 3. An illustration of segment downloading for BEFB scheme

$$T_{Next} \le T_{Use} \quad (1)$$

If the inequality holds, then the client does not download the segment; otherwise the client downloads and stores it to the buffer. The inequality is based on the following reasoning. The broadcasting cycle of segment S_x in this scheme equals 2^i time units if $i<2$ and 2^i+1 time units if $i \ge 2$. Thus, if S_x appears 2^i or 2^i+1 time units before its playing time, then the client does not play the segment during the period, and can wait for the next one. For instance, as indicated in (Figure-3), when client watch segment S_{12} with only diagonal lines on channel C_3 at the second time unit, the client checks the inequality (1), $1+2^3+1 \le 12$, and does not receive the segment. When the client watch next S_{12} with gray color and diagonal lines at the ninth time unit, inequality (1) does not hold, $9+2^3+1>12$, and the client downloads the segment. The client stops loading data from networks when all segments have been received.

Workable Verification: This scheme ensures on-time data delivery on the client side, as explained herein. The papers, [8] and [12] indicates that a video server must broadcast a segment S_x at least once in every x time units to keep on-time data delivery on the client side. The server in this scheme broadcasts segments S_x on channel C_i in every 2^i time units if $i<2$ and 2^i+1 time units if $i \ge 2$. Accordingly, this scheme can ensure continuous video playing on the client side.

IV. Analysis and Comparison

In this section of our literature we will present some analysis of our experiment. Later of this section we will show some comparison with other schemes.

For the analysis, let us consider that we have a video of length, $D=120$ minutes. The rate of data transmission is 10Mbps. If k channels are allocated, then the number of segments in our scheme is 2^k which is higher than in FB and RFB. In FB and RFB, the number of segments is 2^k-1. The length of a segment is $D/2^k$ in our scheme. Both FB and RFB has a segment length $D/(2^k-1)$ which is higher relative to our scheme. When a user requests a video, the client has to wait maximum for the time which is equal to length of the first segment. Hence, our scheme has the lower waiting time.

A comparison of the user waiting time for different number of allocated channels among FB, RFB and BEFB is shown in

TABLE I: Best results Comparison of FB, RFB and BEFB

Criteria	FB	RFB	BEFB
No. of segments	$2^k - 1$	$2^k - 1$	2^k
Length of a segment	$D/2^k - 1$	$D/2^k - 1$	$D/2^k$
Waiting time	$D/2^k - 1$	$D/2^k - 1$	$D/2^k$
Maximum average buffer required	58%	38%	33%

the (Fig. 4).

In FB, for a particular time slot maximum k segments are downloaded. So the client must have the ability to buffer at least k segments. The maximum average buffer required for the FB scheme is 58%. As RFB and BEFB both do not download all the segments available for downloading in a particular time slot, they requires less buffer. RFB requires maximum 38% and BEFB requires 33% of the total video. Here, our scheme, BEFB, out performs FB and RFB. A comparison of maximum buffer requirement in percentage for FB, RFB and BEFB is shown in (Figure-5) using k channels. Table-1 shows some comparative data for FB, RFB and BEFB.

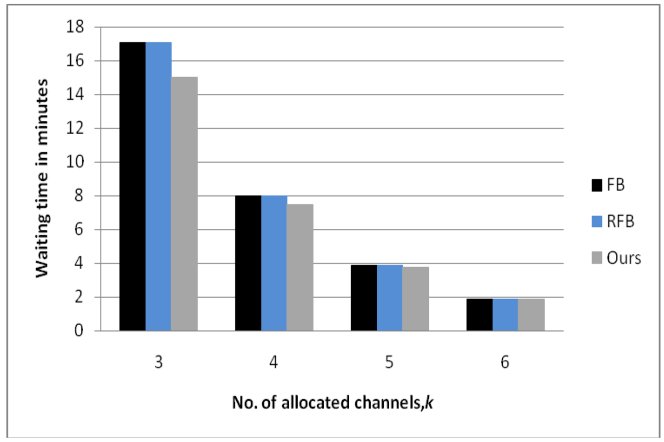

Fig. 4. Comparison of waiting time in minutes for FB, RFB and BEFB using k channels

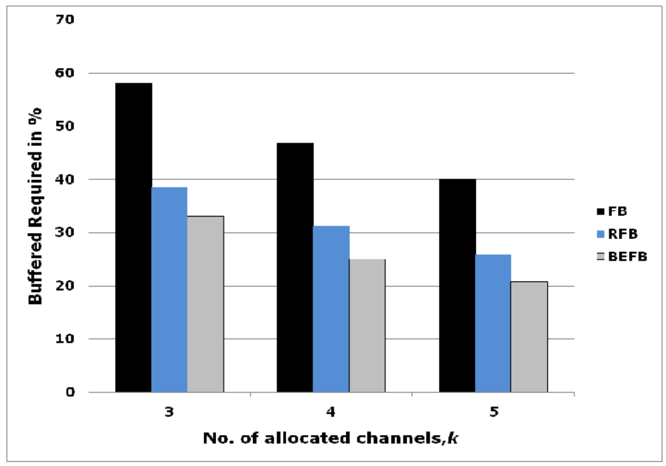

Fig. 5. Comparison of maximum buffer requirement in percentage for FB, RFB and BEFB using k channels

V. Results and Discussion

The Buffer Efficient Fast Broadcasting (BEFB) differs from the Fast Broadcasting scheme (FB) in two ways. First, the number of segments in BEFB is higher than the FB scheme. Second, the BEFB does not download a segment if the segment is downloading useless at that moment. If the segment is broadcasted again at a later time before it is played, then the segment need not to download just now. But, in FB scheme the client has to download un-received segments immediately once they are available for downloading. These two differences lead us to claim the followings:

A. Less User Waiting Time

Both FB and RFB schemes, divides the video into $2^k - 1$ segments. On the other hand, BEFB scheme divides the video into 2^k segments. This reduces the length of a video segment and hence reduces the user waiting time.

B. Reduced Buffer Requirement

The length of a video segment is $D/2^k - 1$ both in FB and RFB; whereas BEFB has a length $D/2^k$. FB downloads all the segments available to download which are not yet downloaded. The average buffer required for FB scheme is 58%. The RFB downloads only the segment which is not transmitted in any slot before the client plays it. If a future segment is transmitted again before the client plays it, the downloading is delayed for the next slot when it is available again to download. This reduces the average buffer requirement for RFB to 38%. In BEFB, it also follows an approach similar to RFB for downloading the segments. As the length of a segment in BEFB is lesser than the RFB, the average buffer requirement is reduced to 33%.

As the number of segments in BEFB is higher than the FB and RFB, the user waiting time is reduced. With this improvement, it also reduces the buffer requirement at the client side which outperforms the RFB scheme.

VI. Conclusion

Most of the Video on Demand Schemes work on user waiting time. FB scheme has smaller waiting time than others. But it is not buffer efficient. RFB proposes a buffer efficient approach for FB. But, our proposed scheme BEFB outperforms RFB by requiring smaller buffer than RFB.

REFERENCES

[1] T. L. Kunii, Y. Shinagawa, R. M. Paul, M. F. Khan, and A. A. Khokhar, "Issues in storage and retrieval of multimedia data", Multimedia Systems, vol. 3, no. 5, pp. 298–304, 1995.

[2] B. Ozden, R. Rastogi, and A. Silberschatz, "On the design of a low cost video-on-demand storage system", Multimedia Systems, vol. 4, no. 1, pp. 40–54, 1996.

[3] A. Dan, D. Sitaram, and P. Shahabuddin, "Dynamic batching policies for an on-demand video server", Multimedia Systems, vol. 4, no. 3, pp. 112–121, June 1996.

[4] K. C. Almeroth and M. H. Ammar, "The use of multicast delivery to provide a scalable and interactive video-on-demand service", IEEE Journal on Selected Areas in Communications, vol. 14, no. 5, pp. 1110–1122, Aug 1996.

[5] S. Viswanathan and T. Imielinski, "Metropolitan area video-on-demand service using pyramid broadcasting", Multimedia Systems, vol.4, pp. 197–208, 1996.

[6] L.-S. Juhn and L.-M. Tseng, "Fast data broadcasting and receiving scheme for popular video services", IEEE Trans. Broadcasting, vol. 44, no. 1, pp. 100–105, March 1998.

[7] ——, "Adaptive fast data broadcasting scheme for video-on-demand services", IEEE Trans. Broadcasting, vol. 44, no. 2, pp. 182–185, June 1998.

[8] Y.-C. Tseng, M.-H. Yang, and C.-H. Chang, "A recursive frequency splitting scheme for broadcasting hot videos in VOD service", IEEE Trans. Communications, vol. 50, no. 8, pp. 1348–1355, August 2002.

[9] A. Bar-Noy and R. E. Ladner, "Windows scheduling problems for broadcast systems", SIAM Journal on Computing, vol. 32, no. 4, pp. 1091–1113, 2003.

[10] J.-P. Sheu, H.-L. Wang, C.-H. Chang, and Y.-C. Tseng, "A fast video-on-demand broadcasting scheme for popular videos", IEEE Trans. Broadcasting, vol. 50, no. 2, pp. 120–125, June 2004.

[11] L.-S. Juhn and L.-M. Tseng, "Harmonic broadcasting for video-on-demand service", IEEE Trans. Broadcasting, vol. 43, no. 3, pp. 268–271, September 1997.

[12] Z.-Y. Yang, L.-S. Juhn, and L.-M. Tseng, "On optimal broadcasting scheme for popular video service", IEEE Trans. Broadcasting, vol. 45, no. 3, pp. 318–322, September 1999.

[13] J.-F. Paris, S. W. Carter, and D. D. E. Long, "Efficient broadcasting protocols for video on demand", in Proceedings of the 6th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems, Montreal, Canada, July 1998, pp. 127–132.

[14] ——, "A low bandwidth broadcasting protocol for video on demand", in Proceedings of the 7th International Conference on Computer Communications and Networks (IC3N'98), Lafayette, LA, October 1998, pp. 690–697.

[15] J.-F. Paris, "A simple low-bandwidth broadcasting protocol for video-on-demand", in Proceedings of International Conference on Computer Communications and Networks, 1999, pp. 118–123.

[16] Z.-Y. Yang, "The Telepresentation System Over Internet With Latecomers Support", Ph.D. Dissertation, Department of Computer Science and Information Engineering, National Central University, Taiwan, 2000.

[17] Hsiang-Fu Yu, Member, Hung-Chang Yang, and Li-Ming Tseng, "Reverse Fast Broadcasting (RFB) for Video-on-Demand Applications", IEEE Trans.on Broadcasting, vol. 53, no. 1, March 2007.

Mohammad Saidur Rahman Earned his B.Sc. in Computer Science & Engineering in 2007 and M.Sc. in Computer Science in 2009 from the American International University– Bangladesh (AIUB). Currently he is working as an assistant professor in the Department of Computer Science, AIUB. His research interests include video on demand, data mining, wireless mesh networks, wireless sensor networks and mobile & multimedia communication.

Mohammad Saiedur Rahaman Completed his B.Sc. in Computer Science & Engineering in 2007 and M.Sc. in Computer Science in 2009 from the American International University– Bangladesh (AIUB). Currently he is working as an assistant professor in the Department of Computer Science, AIUB. His research interests include wireless mesh networks, wireless sensor networks, grid computing, mobile & multimedia communication and data mining.

Ahmed Ridwanul Islam Received the B.Sc. degree in computer science and engineering in 2007 and M.Sc. degree in computer science (Intelligent Systems) in 2009 from American International University-Bangladesh (AIUB). His master's thesis involved object tracing in the presence of partial to full occlusion. He joined AIUB as a lecturer for the department of Computer Sciencein 2008. His research interest includes image processing, virtual reality, human modeling and animation, computer vision, computer games and VoD.

Abhijit Bhowmik Completed his B.Sc. in Computer Science & Engineering in 2009. Now he is pursuing his M.Sc in Computer Science. At the same time he is working a lecturer in the Department of Computer Science, AIUB. His research areas include wireless sensor networks & multimedia communication.