An Efficient Broadcast Scheme for Low Bit-Rate Media Streams

Simon Sheu, Ping-Yi Lin
National Tsing Hua University
sheu@cs.nthu.edu.tw

Mounir A. Tantaoui
University of Central Florida
tantaoui@cs.ucf.edu

Abstract

Periodic Broadcast techniques rely on the server streams sharing for scalable delivery of popular videos. With constant endeavors, several recent studies indicate the server bandwidth utilization can be optimized by the contrived segmentation on each media title. Essentially, a media file needs to be fragmented into numerous smaller segments of precise lengths. This prerequisite is, however, hard to meet in realistic packet-switching networks. Even if a segment can be sized roughly in a few bytes, the inevitable overhead of more induced packet headers becomes significant to deteriorate the bandwidth efficiency. In this paper, we propose a novel broadcast design to preserve the effective bandwidth with no compromise of the original service scalability. Our new design elegantly controls the effect of the overhead on the bandwidth efficiency by only fragmenting the first segment. The latter segments are kept in the constant size as the full-sized packets. As a result, the proposed scheme can avoid the performance deterioration due to the induced overhead of segmentation, while accommodate the available server capacity to reduce the service latency. The performance study indicates our new scheme is constantly better than the existing competitors with all possible design options for the delivery of low bit-rate media streams.

Keywords: Periodic Broadcast, Audio Streaming, Scalability, Efficiency.

1. Introduction

Media streaming is emerging as an increasingly popular content distribution approach for a number of existing and future Internet applications. Periodic Broadcast techniques are one of many such examples [1-4, 6-11, 14-22]. In this paradigm, each media, such video or audio, of length L min. is typically divided into k segments, s_1, s_2, … , s_k. Each s_i is periodically broadcast (or multicast) on the designated channel c_i (e.g., multicast group.) According to s_i's size $|s_i|$ and c_i's capacity $|c_i|$, the packets delivering s_i are repeatedly sent on c_i with a broadcast period proportional to the ratio of $|s_i| / |c_i|$. To support the continuous playback, every broadcast scheme elaborates all the broadcast periods to enable a client arriving at an arbitrary time point to receive each segment no later than its consumption.

Theoretically, in Juhn and Tseng's EHB [10] or Pâris, Carter and Long's PHB [16], each segment can be made as small as the MTU (maximum transmission unit) of the underlying networks. Suppose the packet payload is p bytes and the media playback rate is b Mbps. Then, k can be determined as $\left\lceil \frac{60bL10^6}{8p} \right\rceil$, and s_i needs to be received no later than t = $\frac{8p}{10^6 b}$ sec after the playback of s_{i-1}. If the initial delay is d units of t sec, there will be d+i-1 units for a client upon arrival to complete receiving s_i on c_i. Consequently, its channel capacity should be at least $b/(d+i-1)$ Mbps. The total server bandwidth requirement B can be computed as $\sum_{i=1}^{k} b/(d+i-1) = b(H_{d+k-1} - H_{d-1})$ Mbps, where H_n is the n^{th} harmonic number [9, 16, 23]. A larger value of d will reduce B, as shown in Xu's UEP [23]. Particularly, for a fixed product of d·t, lowering p will also reduce B [10, 16] since halving p will double k and halve t, and thus double d. New $B' = b(H_{2d+2k-1} - H_{2d-1}) = b[(\frac{1}{2d} + \frac{1}{2d+1}) + \ldots + (\frac{1}{2d+2k-2} + \frac{1}{2d+2k-1})] < b[(\frac{1}{d}) + \ldots + (\frac{1}{d+k-1})] = b(H_{d+k-1} - H_{d-1}) = B$. The advantage of using smaller segments and highly précised channel capacity inspires other alternative new designs. Hu's GEBB [5] chose an optimal ratio $r+1 = \sqrt[k]{\frac{L}{dt}+1}$ for $|s_i|/|s_{i-1}|$ and an equal-capacity design $|c_i| = |c_{i-1}| = r \cdot b$. The total bandwidth $B = kb(\sqrt[k]{\frac{L}{dt}+1} - 1)$ improves as k grows. As an example, given $L/(d \cdot t) = 120$, r is reduced from .615 to .0491 and .004807 as k grows from

10 to 100 and 1000. B improves from 6.15b to 4.91b and 4.807b, respectively. The result suggests an infinitely large value of k for optimization. Likewise, Mahanti, *et al.*'s RPB [13] employs a fragmentation scheme for a specified segment streaming rate r·b and requires clients to receive only n out of totally k channels concurrently. The segment size is recursively derived from the following formula:

$$\frac{1}{r}|s_i| = \begin{cases} \frac{1}{r}|s_1| + \sum_{j=1}^{i-1}|s_j| & 1 < i \le n, \\ \sum_{j=k-n}^{k-1}|s_j| & i > n. \end{cases} \quad (1)$$

If n is chosen as k, RPB is exactly the same as GEBB. Otherwise, the client bandwidth requirement can be reduced from B=krb Mbps to C=nrb Mbps. For clear illustration, we plot the latency reduction capability of these schemes in Fig. 1 with earlier designs, where L=120, b=1.5. Both SB (Striping Broadcast) [20] and FB (Fast data Broadcast) [11] use r=1, but their client bandwidth requirements differ: C=nb=3b in SB, while C=kb=B in FB, as indicated by a start (*) in parentheses. Moreover, HB (Harmonic Broadcast) [9] uses d=1, while EHB (Enhanced Harmonic Broadcast) [10] and PHB (Poly-Harmonic Broadcast) [16] both set their parameter d=4 and 1024. Their performance curves are also shown for distinction. Clearly, a larger d will boost the performance to reach the theoretical limit, which can be achieved also by UEP, RPB with r= $1/1024$, n=k, and GEBB with k=1000 or more. As B increases, the initial service latency can be reduced exponentially in all the designs.

The theoretical optimization is built atop the assumption that the segments can be made indefinitely tiny, and even fractional values of segment sizes in bytes are feasible. However, this indispensable prerequisite is hard to meet in realistic networking environments, particularly for delivering low bit-rate media streams. To materialize the idea for audio or voice streaming over IP networks, the packet header overhead will likely pile up an insurmountable barrier. For instance, a three-minute speech of 5 kbps, encoded in G. 723.1 [12], will expect an incoming packet with payload p=1468 bytes every $1468 \cdot 8 / 5 \cdot 1024 \doteq 0.29$ sec. Using RPB with r= $1/1024$ may lead to 2-byte-payload packets per 3.13 ms (or full-sized packet delivery per 2348.8 sec, which will not be an option). Appreciably, the strategy to use very small segments on low-capacity channels does not work. Conversely, over-segmentation to enable a larger k value in broadcast designs induces severe degradation of server bandwidth efficiency. For comparison, we show the performance curves of the schemes with and without taking the packet overheads into consideration in Fig. 2. The left plot, similar to Fig. 1, presents theoretical prediction without considering the overhead. Included with the overhead, those designs that choose smaller segments perform even worse than the simplest design, HB, as shown in the right plot of Fig. 2. Such counter-intuition findings reveal a new challenge for the delivery of low bit-rate audio or voice streams using Periodic Broadcast design. Indeed, the deployment of any broadcast scheme cannot disregard the physical constraints of the underlying networks. In this paper, we investigate the induced overhead of segmentation and develop a new broadcast scheme to preserve the server bandwidth efficiency. In particular, the proposed design also considers the limitation of client bandwidth capacity. Our idea is very simple. However, the new scheme constantly outperforms the existing competitors with all possible design options for the delivery of low bit-rate media streams.

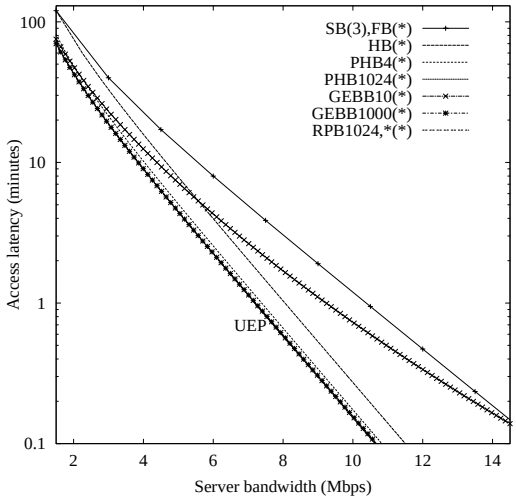


Fig. 1: Comparisons on the latency reduction capability.

The rest of the paper is organized as follows. We introduce the proposed broadcast scheme to minimize the overhead of the segmentation in Section 2. Section 3 presents the performance study. Finally, we give our concluding remarks in Section 4.

2. The Proposed Solution

In presence of the packetization overhead, any broadcast design cannot arbitrarily fragment the media file into an indefinitely number of small segments. Fig. shows PHB with d = 1024 indeed performs much more poorly than PHB with d = 4, which in turn is inferior to PHB with d = 1, namely, HB. Likewise, RPB with r = $\frac{1}{1024}$ is a worse option than RPB with r = $\frac{1}{4}$.

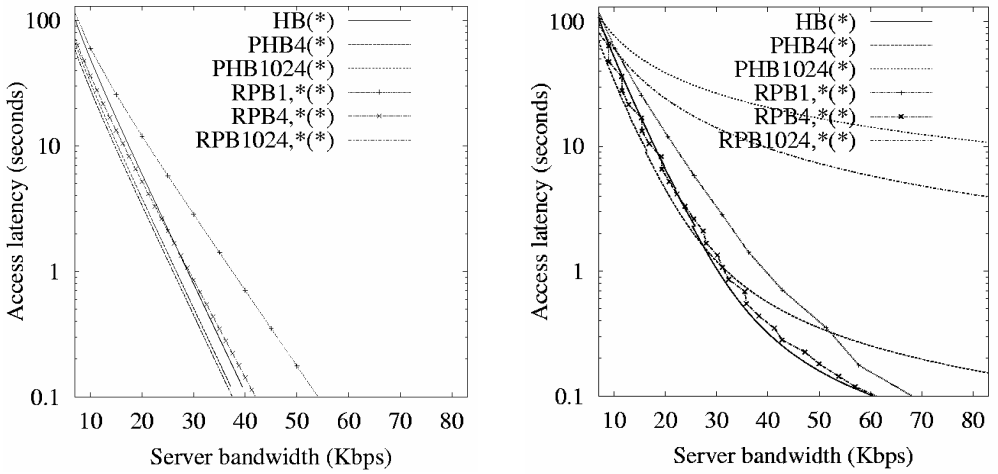


Fig. 2: Effect of packet overheads on broadcast schemes for a 3-min. speech of 5 kbps.

This is a contradictory result to the concept that leads PHB and RPB to reach the theoretical optimization in latency reduction performance. However, HB is not immune from the problem, either. As the server bandwidth B increases, HB also leads to numerous segments of very small size. To overcome this problem, the broadcast design should employ as many full-sized packets as possible. The proposed solution mainly takes advantage of this approach.

We first fragment the media file into a set of m segments, whose sizes are all equal to the packet payload p bytes according to the constraints of the underlying networks. Each segment is designated by its corresponding multicast channel for transmission. We denote T as the time span between two consecutive segments to stream the media file at its playback rate b. That is, $T = \frac{p}{b}$. If the server capacity cannot sustain the latency d in our design to be less than T, we reduce the capacity of channels so as to reduce the bandwidth demand. Surely, d will be equal to or greater than T. Otherwise, we only fragment the first segment into two segments of equal size to make use of this available server capacity. Additional channel is configured to serve the first of two small segments, while the original channel is used to serve the second. Since new segments are smaller, the latency can be improved by utilizing the extra server capacity. We keep this process to repeatedly dichotomize only the very first segment to accommodate the server bandwidth. The inevitable segmentation overhead can be thus confined within the leading segments. The last m-1 segments are kept intact.

Formally, we specify our design by the following mathematic formulation. Each segment s_i is associated with the three parameters: the size $|s_i|$, the receiving start time v_i, and the receiving finish time u_i. These three parameters are used to derive the capacity of the channel c_i. Since s_i is necessarily broadcast on c_i at a period of $u_i - v_i$, the channel capacity $|c_i|$ can be computed as

$$|c_i| = \frac{|s_i|}{u_i - v_i} \times Overhead(s_i), \quad where\ Overhead(s_i) = \frac{|s_i| + \text{Header\_size}}{|s_i|}.$$

(2)

The total bandwidth requirement of our design is subject to the following constraint:

$$\sum_{i=1}^{m} |c_i| \le B : \text{the available server bandwidth capacity.}$$

(3)

These parameters, $|s_i|$, v_i, and u_i, are determined dependent on whether the client bandwidth or processing capacity C is limited or not.

- $C = B$: For clients with unlimited capacity, all the segments can be received upon the arrival of a client. The server bandwidth capacity in serving the media is the same as the client bandwidth capacity.

  In this situation, we first check if B is large enough to make the latency d less than T, the time span between two consecutive segments to stream the media file at its playback rate b:

$$\begin{aligned}\sum_{i=1}^{m} |c_i| &= \sum_{i=1}^{m} \frac{|s_i|}{u_i - v_i} \times Overhead(s_i) \\ &= \sum_{i=1}^{m} \frac{p}{i \cdot T} \times \frac{p + \text{Header\_size}}{p} \\ &= \frac{p + \text{Header\_size}}{T} H_m \le B.\end{aligned}$$

(4)

If the inequality fails, there is no need to further refine the segments since the over-segmentation would only induce more headers, rendering higher bandwidth

demand. Consequently, our design configures m channels out of the available server bandwidth. The capacity of c_i to serve s_i is determined as follows:

$$\begin{cases} d = T \cdot \min_{e} \sum_{i=1}^{m} |c_i| \le B, where\ |c_i| \\ \quad = \frac{p+\text{Header\_size}}{T} \sum_{i=1}^{m} \frac{p}{e+i-1} \\ \quad = \frac{p+\text{Header\_size}}{T} \left(H_{m+e-1} - H_e \right), \\ |s_i| = p, \\ u_i = d + \frac{T}{p} \sum_{j=0}^{i-1} |s_j| = d + T \cdot (i-1), \\ v_i = 0. \end{cases}$$

$$\Rightarrow |c_i| = \frac{|s_i|}{u_i - v_i} \times Overhead\left(s_i\right) = \frac{p}{d+T\cdot(i-1)} \frac{p+\text{Header\_size}}{p} = \frac{p+\text{Header\_size}}{d+T\cdot(i-1)}.$$

(5)

We first determine the minimum value of the latency in multiples of T under the available server bandwidth B. Then, $|c_i|$ is computed in presence of the packet head overhead. Fig. 3 illustrates an example for the playback schedule of some client. Due to the scarcity of server bandwidth, this client needs to wait longer, namely for the latency d, to start the playback in spite of all these m segments being received since arrival. Every segment is sent in terms of a full-sized packet.

If the testing in Eqn. (4) holds, we repeatedly fragment the very first segment into two segments of equal size right before the server bandwidth can no longer support the reduction of the latency. We denote the latency d as $2^{-n}T$ after n iterations since the size of the first two segments is now $\frac{p}{2^n}$. The capacity of c_i to serve s_i can be represented as follows:

$$\begin{cases} |s_i| = \begin{cases} p \cdot 2^{-n}, & i = 1, 2, \\ p \cdot 2^{i-n-2}, & i = 3, \cdots, n+1, \\ p, & i = n+2, \cdots, m+n. \end{cases} \\ u_i = T \cdot 2^{-n} + \frac{T}{p} \sum_{j=0}^{i-1} |s_j| \\ \quad = \begin{cases} T \cdot 2^{-n}(2^i - 1), & i = 1, \cdots, n, \\ T \cdot (i - n + 1 - 2^{-n}), & i = n+1, \cdots, m+n, \end{cases} \\ v_i = 0. \end{cases}$$

$$\Rightarrow |c_i| = \frac{|s_i|}{u_i - v_i} \times Overhead\left(s_i\right).$$

(6)

We note that the total number of segments becomes m+n. The last m-1 segments remain unchanged, while the first n+1 segments s_i, except s_1, is half the size of s_{i+1}. The packet head overhead is fixed regardless of the segment sizes. Fig. 4 shows an example for the playback schedule of some client. Through repeated fragmentation of the very first segment, the latency is the same as the broadcast period and the playout duration of s_1. With n = 2, the original first segment is fragmented twice. At first, two segments of size $\frac{p}{2}$ are created. The first of these two is further dichotomized into two small segments of size $\frac{p}{4}$.

- $C < B$: When the capacity of clients is limited as C, a client is no longer able to receive all the broadcast segments upon arrival. To simplify the design of the client receiving schedule, we employ the following strategy. Given the client capacity C, we accumulate the aggregate capacity

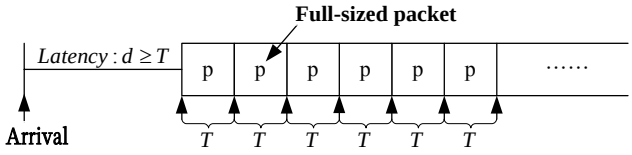


Fig. 3: Playback schedule with latency d ≥ T

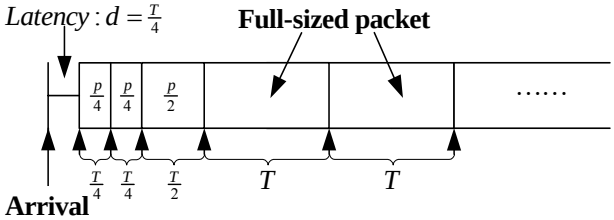


Fig. 4: Playback schedule with latency d < T, n=2.

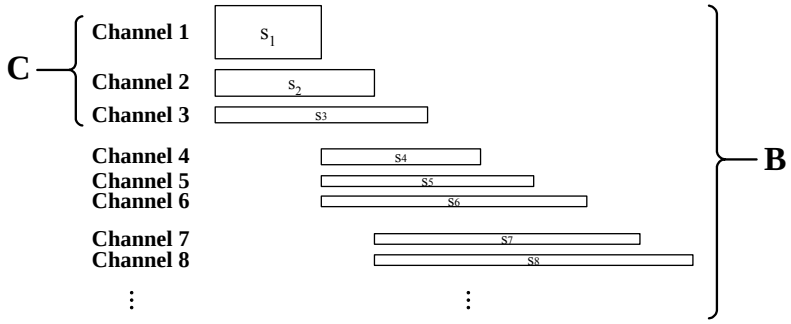


Fig. 5: Receiving strategy for the limited client capacity C < B.

of the first w_1 channels until $\sum_{i=1}^{w_1+1}|c_i| > C$. These w_1 channels are the allowable channels to be received simultaneously since client arrival. After the initial latency, s_1 is completely received. The capacity $|c_1|$ can be reused to initiate the receiving of the next w_2 channels until $\sum_{i=2}^{w_2+1}|c_i| > C$. This process repeats until all the channels are included. To allow each segment to be received in time, the receiving start time v_i of the segment s_i is increased accordingly. The v_i of first w_1 segments to be received upon arrival remains zero. But, the next w_2 segments are updated with their v_i set to u_1, the receiving finish time of the first segment. This update will increase the capacity of their corresponding channels. However, each segment is guaranteed to be received during the scheduled time span from v_i to u_i. Fig. 5 presents one possible receiving scenario of the clients with limited capacity C < B. All the channel capacities amount to B. With the capacity constraint, a client can only receive the first three segments in the beginning. Right after the completion of receiving s_1, the client is then able to receive the next three segments s_4, s_5 & s_6, concurrently with the receiving two earlier segments, s_2 and s_3. Likewise, the completion of receiving s_2 enables s_7 and s_8 to be received.

We employ a numerical procedure to implement the above strategy. Despite lack of the closed form representation for the parameter settings, the procedure can be executed rather fast. The time to generate a feasible schedule subject to the constraints is ignorable compared to the time when the broadcast service is offered using the proposed design.

Comparatively, our new broadcast design resembles PHB and HB, when the inequality testing in Eqn. (4) fails. However, the number of segments in PHB and HB is determined by the ratio of the server bandwidth capacity over the media playback rate. More likely, the segment size is not compatible with the packet payload imposed by the physical networks. As a result, PHB and HB will suffer from the effect of the considerable packet overheads, as shown in Fig. 2. PHB may be able to size the segments to the packet payload artificially. However, as the available server bandwidth increases for the broadcast service, PHB can only improve the service latency by distributing the bandwidth to all the channels. In contrast, our design accommodates the server bandwidth by repeatedly fragmenting the very first segment to effectively reduce the service latency. This strategy makes our solution look more close to the second category of the broadcast designs, such as RPB. In the next section, we present the performance evaluation for these broadcast techniques.

3. Performance Study

To assess the quality of the proposed solution, we perform a series of investigations with the other broadcast schemes. Each media file used in this study is assumed to be 3-minute long with the various playback rates 5 kbps, 6.3 kbps, and 8kbps. It represents the typical audio encoded speeches using G.723.1 (low quality), G. 723.1 (high quality), and G.729 CODEC, respectively [12]. To model the characteristics of the underlying network, we use the packets with the payload p = 1468 bytes plus the 32-byte header. This setting is often applied in typical IP networking environments. For conciseness, HB, PHB, and RPB, discussed earlier in Section 2, are used as to represent the existing broadcast designs. Both PHB and RPB can achieve the theoretical optimization by taking different approaches. For fair comparison, we model the effect of the packet header overheads in these schemes as follows. In HB and PHB, since their segments are of the same size, the induced overhead is easy to compute. Specifically, the realistic server bandwidth requirement is computed as

$$B_{realistic} = \frac{|s_i| + \left\lceil \frac{|s_i|}{p} \right\rceil \times \text{Header\_size}}{|s_i|} \times B.$$

(7)

As for RPB, we first follow the recursive function in Eqn. (1) and the size of the media file to compute $|s_1|$, which is a floating-point value. Then, we take the ceiling of this value as the realistic size of s_1. RPB employs the channels of equal capacity, r·b. Due to the header overheads, the channel capacity is slightly increased to comply with the broadcast schedule in the original design. Mathematically, we denote this realistic value as $r_{realistic}$, which can be computed as:

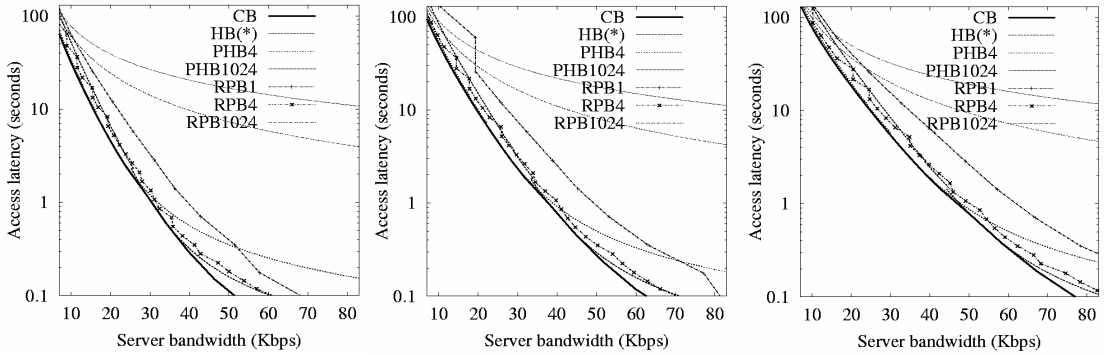
$$r_{realistic} = \frac{\lceil |s_1| \rceil + \left\lceil \frac{\lceil |s_1| \rceil}{p} \right\rceil \times \text{Header\_size}}{\lceil |s_1| \rceil} \times r. \quad (8)$$

Then, we apply Eqn. (1) to obtain the sizes of the other segments recursively. For the sake of in-time delivery, we round down a floating-point value produced each time from Eqn. (1) to an integer value. This value is further used in the subsequent iterations.

3.1: The Effect of Server Bandwidth

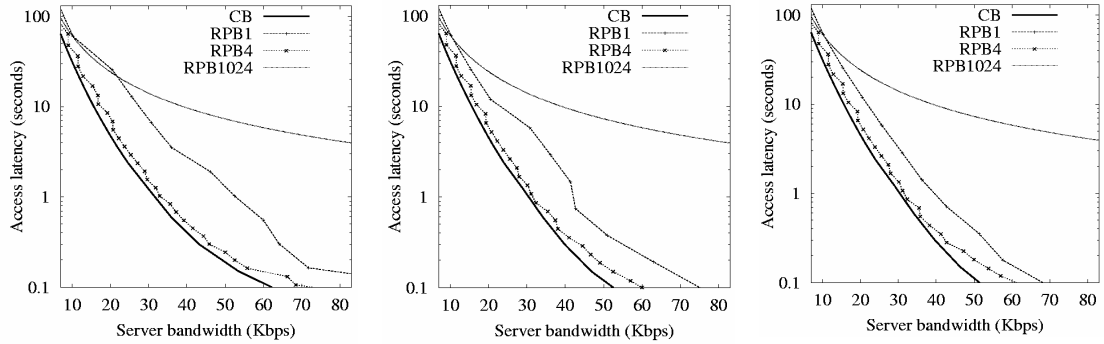
To compare the performance of broadcast schemes, we varied the server bandwidth from 10 kbps to 80 kbps. PHB and RPB with all possible design options are exhaustively examined. The value of the parameter d in PHB can be set from 1 to a very large integer. For brevity, we use d = 1, 4, 1024 to portray the whole PHB family, as denoted by PHB1 (i.e., HB), PHB4, and PHB1024, respectively. Similarly, RPB has the flexibility of setting its parameter r, ranging from 1 down to a very small fraction. We use RPB1, RPB4, RPB1024 to present RPB with r = 1, $\frac{1}{4}$, $\frac{1}{1024}$, respectively. Fig. 6 summarizes the analytical results.

As shown in Fig. 6, all the schemes can reduce the initial access latency as the server bandwidth increase. HB or PHB with d = 1 is the best of its family. A larger d to mince the media file into numerous tiny segments will only degrade its performance in the realistic networking environment. This result is contradictory to its theoretical prediction with no consideration of packet header overheads, as shown in the left graph of Fig. 2. RPB can improve the performance by fine tuning r value. As r reduces to $\frac{1}{1024}$, the performance improves in the beginning, but finally becomes much worse with very small r. The idea to use smaller segments does not always work. In particular, only the exhaustive search algorithm can determine the best parameter settings for PHB and RPB. In contrast, our current broadcast design, denoted as CB, can automatically adapt to the working environment. It only fragments the very first segment to accommodate the available server capacity. As shown in Fig. 6(a), CB outperforms PHB and



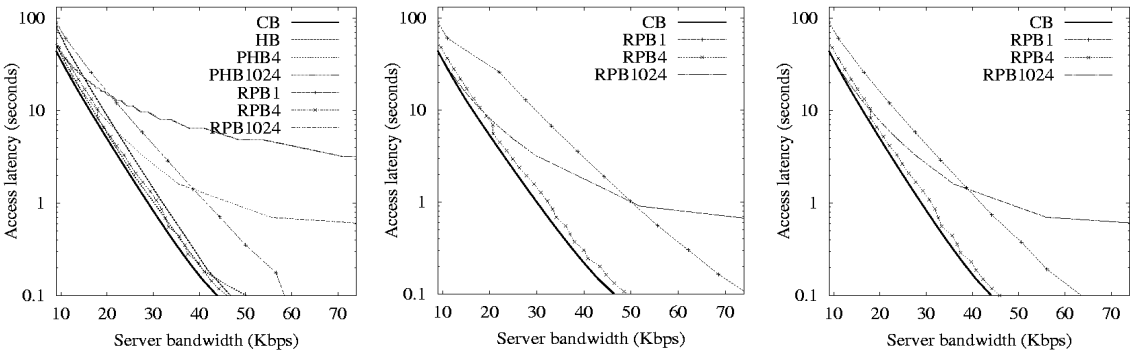
(a) Playback rate b = 5 kbps. (b) Playback rate b = 6.3 kbps. (c) Playback rate b = 8 kbps.

Fig. 6: The effect of server bandwidth for the media file with different playback rates.



(a) Client capacity C = 15 kbps. (b) Client capacity C = 25 kbps. (c) Client capacity C = 40 kbps.

Fig. 7: The effect of client bandwidth.



(a)Client capacity C is not limited (b)Client capacity C = 15kbps (c)Client capacity C = 25 kbps

Fig. 8: Performance evaluation of ATM protocol

RPB even with their best options. As the server bandwidth increases to 40 kbps or more, the significant performance improvement is achievable by CB. We also repeat the experiments for different playback rates. Fig. 6(b) & (c) demonstrate the consistent results. The proposed broadcast scheme is the best performer.

3.2: The Effect of Client Bandwidth

In this study, we investigate the situation that the bandwidth or processing capacity of clients is limited. PHB and HB do not support the clients with limited bandwidth. We only compare CB with RPB in the experiments. Fig. 7 presents the results. When the client capacity is limited to 15 kbps, the proposed scheme is constantly superior to RPB with all design choices, as shown in Fig. 7(a). RPB1024 suffers considerably from over-segmentation. Delivering the extremely tiny segments over numerous channels will only make its performance even worse. On the other hand, RPB4 does outperform RPB1 by taking advantage of finer segments. The best setting of the parameter r can be obtained only through exhaustive testing. In particular, r is a floating-point value, rendering finding the best value difficult. By our simple design, CB can avoid this difficulty. It exploits the characteristics of the underlying networks to fragment the media file into segments of the size that is equal to the payload size of the packets. By such feature, CB preserves the server bandwidth efficiency. As more server capacity available for the broadcast service, CB repeatedly fragment only the very first segment to best utilize the bandwidth. Fig. 7 (b) & (c) show the results of performance evaluation when C is limited to 25 and 40 kbps, respectively. Both also indicate CB can outperform RPB by a considerable margin over a wide range of server capacity.

3.3: Comparisons with ATM protocol

As shown in Table 1, header and payload sizes are different between IP Network and ATM protocol. However, the results of performance evaluation in ATM are similar to IP Network, shown in Fig. 8.

	Header	Payload
IP Network	32 bytes	1468 bytes
ATM	5 bytes	48 bytes

Table 1

Fig. 8(a) shows when client capacity C is not limited, the proposed broadcast scheme is the best performer. Fig. 8(b) & (c) show the results of performance evaluation when C is limited to 25 and 40 kbps. They also demonstrate the consistent results.

4. Concluding Remarks

Theoretical optimization of broadcast designs is possible only if the delivery of segments incurs no transmission overhead. In particular, indefinitely small or fractionally precise values must be allowed for the segment sizes. However, the idea is not attainable. In reality, such approach suffers from the packet header overheads due to over-segmentation in practical working environments. In this paper, we investigate this problem and develop a solution by taking the networking constraints into consideration. Strategically, our scheme fragments only the very first segment to exploit the server capacity, while keeping the last segments in the constant size as the full-sized packets. As a result, the proposed design can achieve the best service scalability in terms of initial latency reduction while maximizing the server bandwidth efficiency.

Acknowledgements

We are devoutly thankful for the supports of the National Science Council, Project No. NSC 91-2622-E-007-034-CC3, and the MOE Program for Promoting Academic Excellent of Universities, Grant No. 89-E-FA04-1-4, Taiwan, R.O.C.

Reference

[1] M. K. Bradshaw, B. Wang, S. Sen, L. Gao, J. Kurose, P. Shenoy, and D. Towsley, "Periodic Broadcast and Patching Services - Implementation, Measurement, and Analysis in an Internet Streaming Video Testbed," in Proc. of ACM Multimedia, Ottawa, Ontario, Canada, Sept., 2001.

[2] Y. Cai, K. A. Hua, and S. Sheu, "Leverage Client Bandwidth to Improve Service Latency in a Periodic Broadcast Environment," Journal of Applied Systems Studies (special issue), 2(3), 2001.

[3] S. R. Carter and J.-F. Pâris, "A Dynamic Heuristic Broadcasting Protocol for Video-on-Demand," in Proc. of IEEE INFOCOM, pp. 657-664, 2001.

[4] D. L. Eager and M. K. Vernon, "Dynamic Skyscraper Broadcasts for Video-on-Demand," in Proc. of the Int'l Workshop on Multimedia Information Systems, Turkey, pp. 18-32, Sept., 1998.

[5] A. Hu, "Video-on-Demand Broadcasting Protocols: A Comprehensive Study," in Proc. of IEEE INFOCOM, 2001.

[6] K. A. Hua, Y. Cai, and S. Sheu, "Exploiting Client Bandwidth for more Efficient Video Broadcast," in Proc. of the Int'l Conf. on Computer, Communications and Networks, Lafayette, Louisiana, Oct., 1998.

[7] K. A. Hua and S. Sheu, "Skyscraper Broadcasting: A New Broadcasting Scheme for Metropolitan Video-on-Demand Systems," Proc. of ACM SIGCOMM, Computer Communication Review, 27(4):89-100, Sept., 1997.

[8] K. A. Hua and S. Sheu, "An Efficient Periodic Broadcast Technique for Digital Video Libraries," Multimedia Tools and Applications, 10(2/3):157-177, April, 2000.

[9] L.-S. Juhn and L.-M. Tseng, "Harmonic Broadcasting for Video-on-Demand Service," IEEE Trans. on Broadcasting, 43(3):268-271, Sept., 1997.

[10] L.-S. Juhn and L.-M. Tseng, "Enhanced Harmonic Data Broadcasting and Receiving Scheme for Popular Video Service," IEEE Trans. on Consumer Electronics, 44(2):343-346, May, 1998.

[11] L.-S. Juhn and L.-M. Tseng, "Fast Data Broadcasting and Receiving Scheme for Popular Video Service," IEEE Transaction on Broadcasting, 44(1):100-105, March, 1998.

[12] V. Kumar, M. Korpi, and S. Sengodan, IP Telephony with H.323: Architecture for Unified Networks and Integrated Services: Wiley Computer Publishing, 2001.

[13] A. Mahanti, D. L. Eager, M. K. Vernon, and D. Sundaram-Stukel, "Scalable On-Demand Media Streaming with Packet Loss Recovery," in Proc. of ACM SIGCOMM, San Diego, CA, pp. 97-108, August, 2001.

[14] J.-F. Pâris, "A Simple Low-Bandwidth Broadcasting Protocol for Video-on-Demand," in Proc. of the Int'l Conf. on Computer, Communications and Networks, pp. 690-697, Oct., 1999.

[15] J.-F. Pâris, S. W. Carter, and D. D. E. Long, "Efficient Broadcasting Protocols for Video-on-Demand," in Proc. of the 6th Int'l Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS '98), pp. 127-132, July, 1998.

[16] J.-F. Pâris, S. W. Carter, and D. D. E. Long, "A Low Bandwidth Broadcasting Protocol for Video-on-Demand," in Proc. of the Int'l Conf. on Computer, Communications and Networks, Lafayette, Louisiana, Oct., 1998.

[17] J.-F. Pâris, S. W. Carter, and D. D. E. Long, "A Hybrid Broadcasting Protocol for Video-on-Demand," in Proc. of SPIE Multimedia Computing and Networking, pp. 317-326, Jan., 1999.

[18] J.-F. Pâris, S. W. Carter, and D. D. E. Long, "A Universal Distribution Protocol for Video-on-Demand," in Proc. of IEEE International Conference on Multimedia and Expo, New York, NY, USA, pp. 49-52, 2000.

[19] J.-F. Pâris, D. D. E. Long, and P. E. Mantey, "Zero-Delay Broadcasting Protocols for Video-on-Demand," in Proc. of ACM Multimedia, Orlando, pp. 189-197, 1999.

[20] S. Sheu, K. A. Hua, and Y. Cai, "A Novel Broadcast Technique for Theaters in the Air," in Proc. of Workshop on Virtual University for Multilingual Education, Chicago, IL, pp. 218-225, July, 2000.

[21] Y. C. Tseng, C. M. Hsieh, M. H. Yang, W. H. Laio, and J. P. Sheu, "Data Broadcasting and Seamless Channel Transition for Highly-Demanded Vodeos," in Proc. of IEEE INFOCOM, pp. 727-736, 2000.

[22] S. Viswanathan and T. Imielinski, "Pyramid Broadcasting for Video-on-Demand Service," in Proc. of the SPIE Multimedia Computing and Networking, pp. 66-77, Feb., 1995.

[23] L. Xu, "Efficient and Scalable On-Demand Data Streaming Using UEP Codes," in Proc. of ACM Multimedia, Ottawa, Ontario, Canada, pp., Sept., 2001.