

Adaptive Live Broadcasting for Highly-Demand Videos*

HUNG-CHANG YANG, HSIANG-FU YU AND LI-MING TSENG

Distributed System Laboratory

Department of Computer Science and Information Engineering

National Central University

Chungli, 320 Taiwan

E-mail: {cyht; yu}@dslab.csie.ncu.edu.tw

E-mail: tsenglm@csie.ncu.edu.tw

With the growth of broadband networks, Video-on-Demand (VoD) has become realistic. Many significant broadcasting schemes have been proposed to reduce the bandwidth requirements for stored popular videos, but they cannot be used to support live video broadcast perfectly. Herein, we propose a new broadcasting scheme, called the *Adaptive Live Broadcasting (ALB)* scheme, which supports live video broadcasting and performs well over a wide range of request arrival rates. From our analysis and comparison, we find that our ALB scheme is suitable for broadcasting live video. It has several significant advantages: (1) It has the shortest maximum waiting time with fixed channels. (2) It has the lowest maximum I/O transfer requirements with a fixed maximum waiting time at the client end. A simulation is employed to evaluate several live broadcasting schemes: UD, ST, AFB and ALB. The results reveal that our ALB scheme consumes the least server bandwidth.

Keywords: adaptive live broadcasting scheme, network bandwidth scheduling, popular video service, video-on-demand (VoD), multimedia systems

1. INTRODUCTION

With the growth of broadband networks, Video-on-Demand (VoD) [13] has become realistic. Many studies have investigated VoD. One of the important areas of research is the issue of how to distribute the top ten or twenty so-called “hot” videos more efficiently. Broadcasting is a promising solution. It transfers each video according to a fixed schedule and consumes a constant bandwidth regardless of the number of requests for that video. That is, the number of users watching a given video is independent of their bandwidth requirements. A basic broadcasting scheme is the batch scheme [1]. The batch scheme delays the users’ requests for a certain amount of time and serves these requests in a batch so that bandwidth consumption is reduced. However, the batch scheme still requires quite a large bandwidth for a hot video. For example, for a film that lasts 120 minutes, if each request for the film has to be served within 10 minutes, then we need to allocate 12 (120/10) video channels.

Suppose a set-top-box (STB) at the client end can buffer portions of the playing video on disk. With the STB, there are many available broadcasting schemes, such as

Received May 15, 2002; accepted July 25, 2002.

Communicated by Biing-Feng Wang, Stephan Olariu and Gen-Huey Chen.

* The work was supported in part by a research grant from the National Science Council of the ROC under contract number NSC 90-2213-E-008-049. A preliminary version of the paper was presented at the 2002 International Conference on Parallel and Distributed Systems, Chungli, Taiwan.

fast broadcasting (FB) [4, 8], pagoda broadcasting (PB) [10], new pagoda broadcasting (NPB) [11], recursive frequency splitting (RFS) [3], staircase broadcasting (SB) [6] and harmonic broadcasting (HB) [5, 7]. These schemes divide a video into multiple equal-size segments and distribute these segments through several independent data streams. In addition, they require the STB to receive all the segments from the streams when the user starts to watch the video. These broadcasting schemes substantially reduce the bandwidth requirements for hot videos. For example, with the FB, a video server allocates 4 video streams for a 120-minute video, and its waiting time is less than 8 minutes. Both the bandwidth consumption and waiting time of the fast broadcasting scheme are superior to those of the batch scheme.

In the real world, some historical events are very hot, for example, the collision of the comet Shoemaker-Levy with Jupiter, and thousands of people attempt to connect to the Internet to simultaneously watch the video. Such actions easily produce network congestion. However, most of these schemes, such as PB, NPB, RFS, SB and HB, can not broadcast such hot live videos and alleviate congestion. In order to overcome this obstacle, this paper proposes the Adaptive Live Broadcasting (ALB) scheme, which supports live video broadcasting. The simulation results indicate that ALB has a shorter waiting time and lower bandwidth requirements than 4 other live broadcasting schemes.

The rest of this paper is organized as follows. The next section introduces related works. Section 3 describes the ALB. Section 4 provides analysis and a comparison. Section 5 presents the simulation results. Conclusions are drawn in section 6.

2. RELATED WORK

2.1 The Requirements of Live Video Broadcasting

Initially, we will analyze three important requirements for live video broadcasting that differs from those of stored video broadcasting.

- R1. *The total data transfer rate can not be larger than the media production rate.* In the case of live broadcasting, new media is produced at a constant speed, so broadcasting schemes that always transfer data at a rate higher than the media production rate can not support live broadcasting.
- R2. *A live video segment can not be transferred preemptively until the video segment is produced.* The scenes in a live video are captured and broadcast with video progress; the broadcasting schemes can not transmit posterior and unavailable segments of live video in advance.
- R3. *A broadcasting scheme has to tolerate the variation the lengths of live videos.* People always hope that a live video will run according to schedule; however, in the real world, a live video often ends either early or late, rarely on time. Most broadcasting schemes assume that the video's length is known and fixed. If the video ends early, the broadcasting schemes simply free up the allocated channels or repeat the last or blank video segments. Hence, the viewer watching the video is not affected. If the video ends late, the broadcasting schemes require additional bandwidth to handle this situation.

2.2 The Schemes Regarding Live Broadcasting

2.2.1 New Pagoda Broadcasting scheme (NPB)

NPB [11] employs a *rectangular matrix allocation* method to distribute the segment-to-stream mapping. The mapping is optimal when each segment S_i can be broadcast exactly once every i slots. Accordingly, as far as possible NPB broadcasts each segment S_i once every i slots.

Fig. 1 depicts NPB with 4 streams. It is able to transmit 26 segments, and it guarantees that the viewing delay will not exceed 4 minutes, 37 seconds for a 2-hour video.

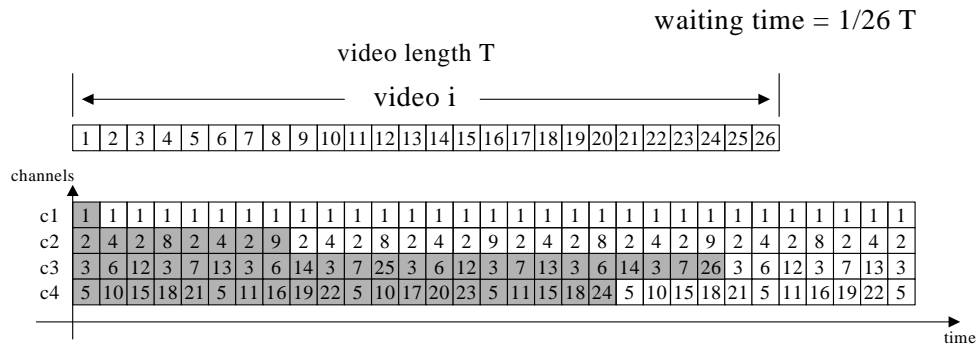


Fig. 1. The NPB scheme with 4 streams.

When we attempt to apply NPB to live video broadcasting, we find that it does not satisfy requirements R1 and R2. For example, broadcasting a video with 3 streams as shown in Fig. 2. The segments S_2 and S_3 are unavailable at slot 1, segments S_4 and S_6 are unavailable at slot 2, and so on.

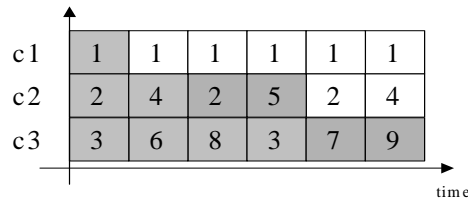


Fig. 2. The NPB scheme with 3 streams.

If we add an additional live stream and delay the distribution of some segments, the NPB can broadcast the live video. We call this the *Live NPB* scheme, and it is shown in Fig. 3.

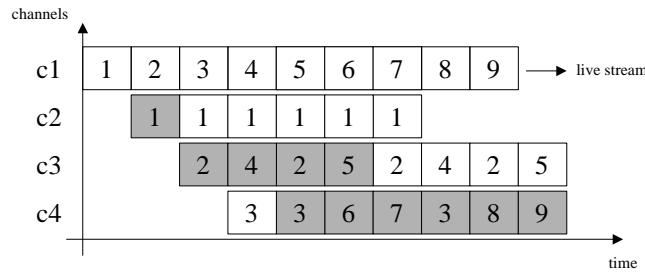


Fig. 3. The Live NPB scheme with 4 streams.

2.2.2 Recursive Frequency Splitting scheme (RFS)

By using a more complex segment-to-stream mapping, the RFS [6] provides a shorter waiting time than the NPB scheme when the number of streams is larger than 4. Fig. 4 depicts the RFS with 4 streams.

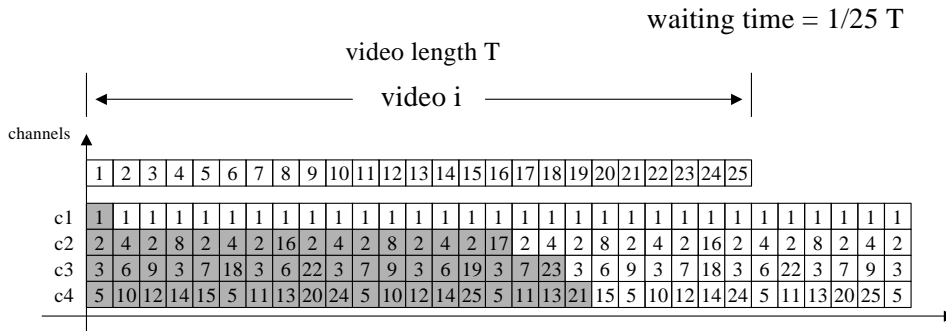


Fig. 4. The RFS scheme with 4 streams.

The RFS also fails to satisfy requirements R1 and R2. We must add an additional live stream and delay the distribution of some segments to support live video broadcasting. This is the called *Live RFS* scheme.

2.2.3 Fast Broadcasting scheme (FB)

The FB scheme [4, 8] reduces the bandwidth requirement in the logarithmic order of the maximum waiting time. It partitions the video into $2^k - 1$ segments, S_1 to S_{2^k-1} , and stream j , where $1 \leq j \leq k$, transmits segments S_2^{j-1} to $S_{2^j-1}^j$ as indicated in Fig. 5.

The FB scheme can directly support live video broadcasting with slight modification as shown in Fig. 6. As shown in Fig. 6(a), the FB scheme can support live video broadcasting by delaying the distribution of segments. Fig. 6(b) shows the *Live FB* scheme.

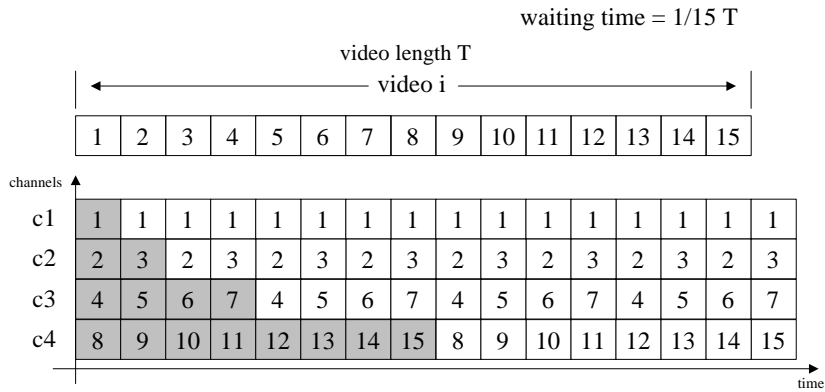
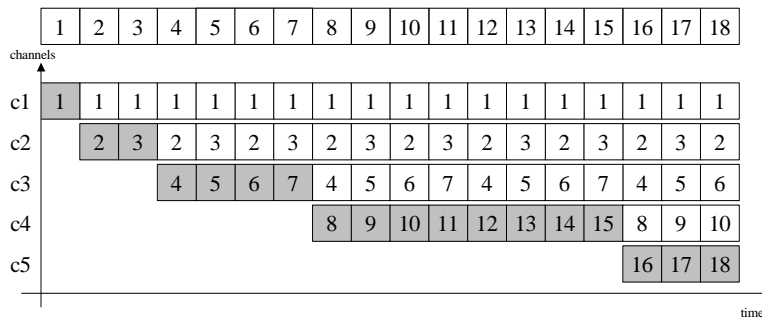
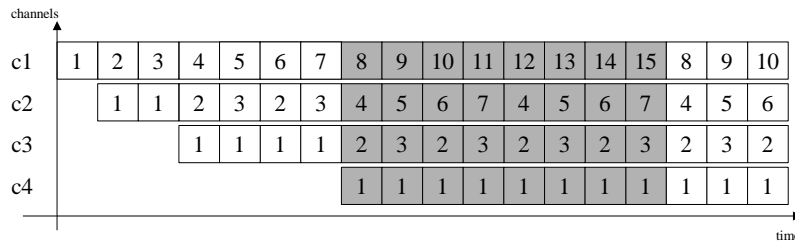


Fig. 5 Illustration of channel allocation for FB.



(a)



(b)

Fig. 6. The Live FB scheme.

2.2.4 Adaptive Fast Broadcasting scheme (AFB)

The disadvantage of FB is that it can not dynamically allocate bandwidth even though no requests arrive. To overcome this obstacle, the AFB scheme [9] dynamically allocates bandwidth according to the users' requests.

The AFB scheme can also support live video broadcasting because it is based on FB. An illustration of the AFB scheme is shown as Fig. 7, where it is assumed $N = 15$, and that there are 3 requests.

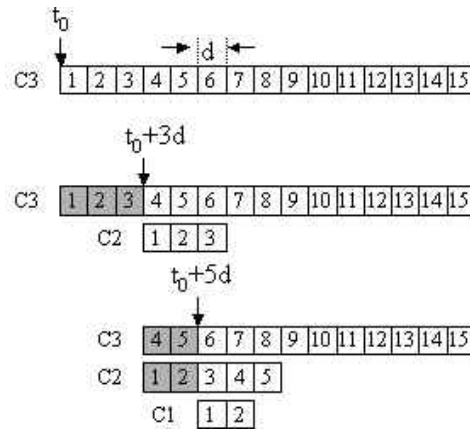


Fig. 7. The AFB scheme.

2.2.5 Universal Distribution scheme (UD)

The universal distribution scheme [12] is a dynamic broadcasting scheme based on the FB scheme. In Fig. 8, the first request arrives at slot 0, and it is followed by two other requests that arrive at slots 3 and 4, respectively. Note that segments S_2 and S_3 are allocated to slots 4 and 5, but that they are allocated to slots 5 and 6. Hence, segments S_2 and S_3 can be shared with the second and third request.

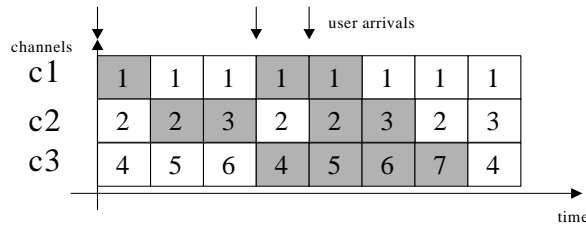


Fig. 8. The UD scheme with 3 requests.

Since UD is based upon FB, it also supports live video broadcasting, as shown in Fig. 9, and we call it the *Live UD* scheme.

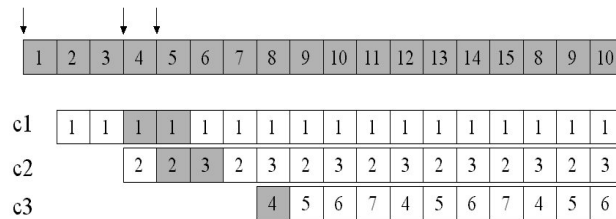


Fig. 9. The Live UD scheme.

2.2.6 Stream Tapping scheme (ST)

ST [2] assumes that clients have a small buffer on their STB. The buffer allows them to “tap” into streams of data on the video server originally created for other clients, and to then store the data until they are needed. In the best case, clients can get most of their data from existing streams, which greatly reduces the duration of their own stream. The ST scheme can support live video broadcasting.

3. ADAPTIVE LIVE BROADCASTING SCHEME

In this section, we propose a new broadcasting scheme, called the *adaptive live broadcasting* (ALB) scheme, to support live video broadcasting.

Before getting into the details of our algorithm, we give the following necessary lemmas. Assume the number of segments of live video is n , and that the number of requests is m .

Lemma 1. Each segment S_i , $1 \leq i \leq n$, must be broadcast at least once every i consecutive time slots on one of the k channels.

Proof: Suppose that a user starts playing the video at time slot j . Then the user will consume segment S_i at time slot $j + i - 1$. This implies that S_i must be broadcast on one of the channels at time slot $j + i - 1$, or has been broadcast on one of the channels during slots $j, j + 1 \dots j + i - 2$. This proves that the condition given in the lemma is a sufficient condition. Next, we will prove that this is also a necessary condition. If S_i has not been sent on the aforementioned time slots, the user will experience an interruption at time slot $j + i - 1$.

Lemma 2. To support live video broadcasting, each segment S_i , $1 \leq i \leq n$, must be broadcast in time slot i for the first time.

Proof: Since we cannot pre-fetch the live video and store it in the disk of the VoD server beforehand, each segment S_i must be broadcast at time slot i for the first time.

Lemma 3. The necessary recasting segments N_i for request R_i that arrives at time slot T_i are $BS_{T_i} - BS_{T_i-1}$, where $1 \leq i \leq m$ and BS_{T_i} are the segments broadcast from time slot 0 to time slot T_i .

As shown in Fig. 10, when the second user enters into the session at the 9th slot, the video server has to recast segment $N_2 = BS_9 - BS_6 = \{S_1, S_2, S_3, S_7, S_8 \text{ and } S_9\}$ according to lemma 3, when the fourth user enters the session at the 14th slot, the video server has to recast segments $N_4 = BS_{14} - BS_{11} = \{S_1, S_2, S_3, S_4, S_6, S_7, S_8, S_{12}, S_{13} \text{ and } S_{14}\}$, and so on.

In the following, we introduce our adaptive live broadcasting scheme. In order to satisfy Lemma 1, we put segment S_i in every i time slots. If there is no free slot at that position, segment S_i is put into the previous time slot until it can be. To satisfy Lemma 2, we use a stream, called a live stream, to broadcast segments $S_1, S_2, S_3 \dots S_n$. Fig. 11 illustrates the segment to stream mapping.

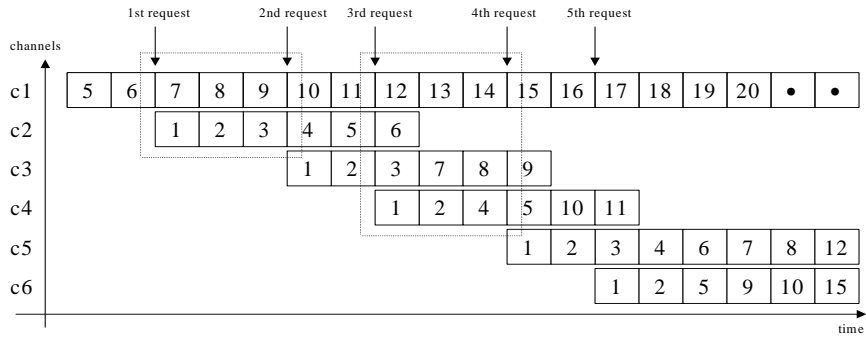


Fig. 10. The necessary recasting segments when the user enters the session.

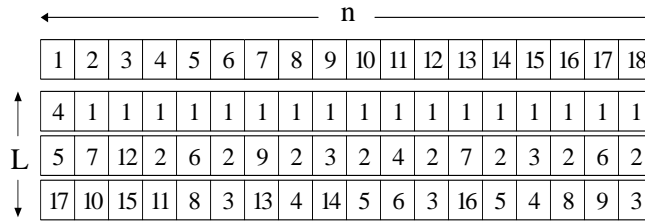


Fig. 11. The segment to stream mapping.

Fig. 12 presents the entire algorithm of the adaptive live broadcasting scheme.

Assumptions:

- the number of segments is n
- L indicates the total number of channels minus 1
- slot k contains m_k segments

Algorithm:

```

initialize all  $m_k$  to 0
for  $i := 1$  to  $n$  do
  if  $n \bmod i$  equal 0 then  $max = \left\lceil \frac{n}{i} \right\rceil + 1$ 
  else  $max = \left\lceil \frac{n}{i} \right\rceil$ 
  for  $j := 2$  to  $max$  do
     $p = i*j$ 
    while  $m_p \leq L$  do
       $p = p - 1$ 
    end while
    if  $p > n$  then  $p = p - n$ 
    schedule  $S_i$  in slot  $p$ 
     $m_p = m_p + 1$ 
  end for loop
end for loop

```

Fig. 12. The entire ALB algorithm.

So far, our proposed scheme can satisfy requirements R1 and R2. In the following, we will propose two approaches to satisfying requirement R3.

The first approach discards the excess part of the video. The video is like an hour of news on CNN. The older news is discarded, and newer news is added. Our adaptive live broadcasting scheme can broadcast an hour of news in a period and broadcast newer news by discarding older news at the next period, as shown in Fig. 13. The period has seven time slots. In the first hour, we broadcast news 1 to 7. In the second hour, we broadcast news 2 to 8 by discarding the older news, 1, and adding the newer news, 8.

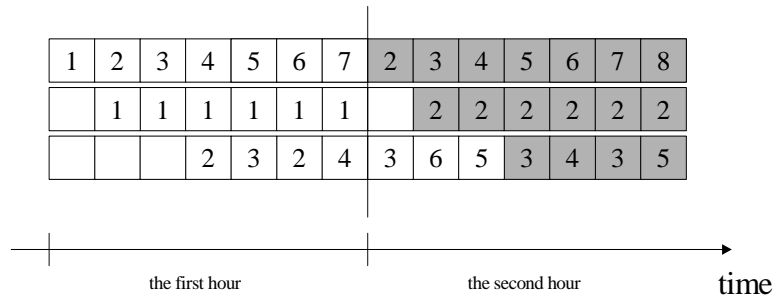


Fig. 13. An example of broadcasting an hour of news.

The second approach allocates additional channels to transfer excessively long segments of live video, as shown in Fig. 14.

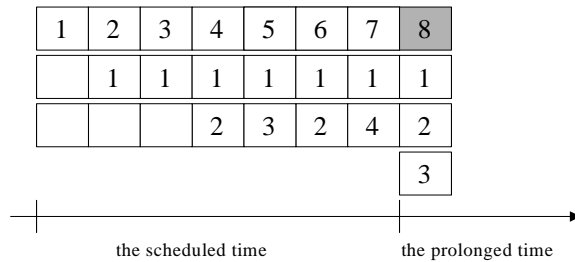


Fig. 14. Broadcasting excessively long segments by allocating additional channels.

To efficiently play videos using ALB, the user's request are served on demand based on the following principles:

- 1) We exploit the live stream as the main stream and only recast necessary segments when the user enters the session.
- 2) A later user can share segments that are rebroadcast to previous users.
- 3) The necessary recasting segments S_i delay i time slots to broadcast as possible as it can when the user enters into session.

Fig. 15 illustrates the use of ALB. The first channel is the live stream, and 4 users enter the session. When a user enters the session at the 2nd time slot, the video server has to rebroadcast segments S_1 and S_2 . When the user enters the session at 3rd time slot, the video server has to rebroadcast segments S_1 and S_3 , because S_2 has been rebroadcast. When a user enters the session at the 9th time slot, the video server has to rebroadcast segments $S_1, S_2, S_3 \dots S_9$. As indicated in Fig. 15, segment S_5 is rebroadcast at the 14th time slot, not the 10th time slot. This is because distribution must be delayed as long as possible to increase the probability of sharing the bandwidth. Because the user's request is served on demand, the ALB scheme requires less bandwidth.

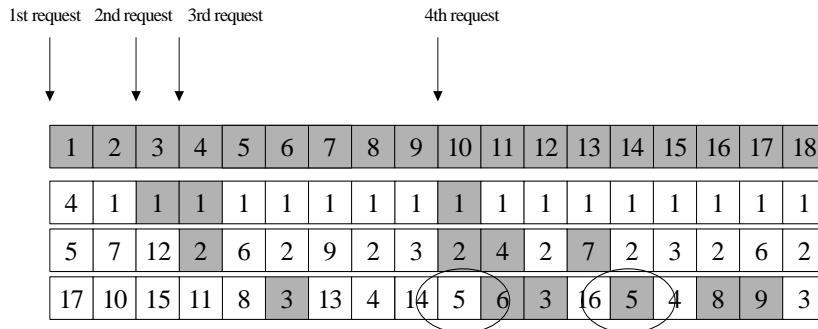


Fig. 15. Efficiently playing video using ALB.

4. ANALYSIS AND COMPARISON

First, we will derive the maximum number of segments with fixed channels and then compare the results obtained using ALB with those obtained using some existing schemes. We will also analyze the user waiting time and disk rate transfer requirement.

4.1 Maximum Segments With Fixed Channels

To maximize bandwidth utilization, we need to obtain the maximum degree of sharing of each rebroadcast segment. In order to achieve the goal, the scheme must delay segment distribution as long as possible. According to Lemma 1, if a transmission schedule starting at slot $i + 1$ cannot share its j -th segment S_j with any previous transmission schedule, then the schedule will attempt to put segment S_j in slot $i + j$. Thus, the first segment must be scheduled at least once every slots, the second segment must be scheduled at least once every two slots, and so on. Therefore, we can distribute each segment on demand according to its minimum frequency. The algorithm is shown in Fig. 16.

In the worst case, there is at least one user at each time slot, and the segments that user requires are plotted in Fig. 17.

In addition, we can calculate the maximum number of segments, n , with fixing channels, c , as follows:

Assumptions:

a new video request arrives during slot i

Algorithm:

```

for  $j := 1$  to  $i$  do
    search slots  $i + 1$  to  $i + j$  for an already scheduled segment of  $S_j$ 
    if not found then
        schedule segment  $S_j$  in slot  $i + j$ 
    end if
end for loop
    
```

Fig. 16. The algorithm for minimum frequency scheduling.

live segments																																									
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40		
recast segments																																									
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
		2		2		2	3	2		2		2	3	2		2		2	3	2		2		2	5	2	3	2		2		2	3	2	5	2		2	3	2	
			3		4		5		3		7	5	4		3		4	7	11		3		13	9	4		3		4	11	17	7	3		19	13	4				
								4				8		6		5				4				7		5		8				4				6			5		
							6						9	10						6				14		6		16				6							8		
																				8							10									9			10		
																					12						15									12			20		
																																				18					

Fig. 17. The segments that user requires in the worst case.

$$\sum_{i=1}^{n-1} (\text{number of factors of } i) + n \leq n * c$$

Next, we will show how to calculate the maximum number of segments for the ALB scheme. The minimum number of segments to be broadcast is $\sum_{i=1}^n \left\lceil \frac{n}{i} \right\rceil$ since segment S_i must be broadcast every i time slots. Furthermore, we consider that all the segments whose index is a factor of the value n cannot be scheduled in slot n due to the fixed channels. Therefore, the up-bound can be calculated by as follows:

$$\sum_{i=1}^n \left\lceil \frac{n}{i} \right\rceil + [(\text{number of factors of } n) - c] \leq c * n .$$

Finally, we derive the maximum segments with fixed channels from Live RFS, Live NPB, Live FB and ALB schemes. Fig. 18 lists the result, where its first row indicates the number of channels. We can find that the ALB scheme outperforms all other schemes.

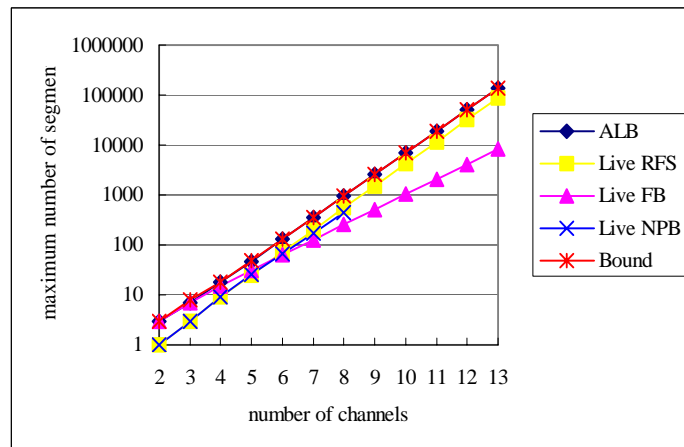
4.2 Waiting Time vs. Bandwidth Allocation

Suppose the client end has enough disk space to buffer portions of the video on the disk. When we miss a segment, S_j , of a requested video, the maximum waiting time will

be equal to the access time of S_j . The length of the video is D , which is equally divided into N data segments. Therefore, using the ALB scheme, the maximum waiting time for accessing a broadcast video is $\frac{D}{N}$. Fig. 19 shows maximum waiting time vs. network bandwidth allocation for the ALB, Live RFS, Live FB and Live NPB schemes. For example, when the number of channels is 13, the waiting time we obtained using the ALB scheme is 17 times shorter than the waiting time obtained using the Live FB scheme.

Channel	2	3	4	5	6	7	8	9	10	11	12	13
Bound	3	8	18	48	130	350	936	2550	6952	18876	51300	139464
ALB	3	7	18	47	129	349	935	2549	6951	18875	51299	139463
Live RFS	1	3	9	25	73	201	565	1522	4284	11637	31677	86428
Live FB	3	7	15	31	63	127	255	511	1023	2047	4095	8191
Live NPB	1	3	9	26	66	172	442	N/A	N/A	N/A	N/A	N/A

(a)



(b)

Fig. 18. The maximum number of segments in different schemes.

In Fig. 20, we compare the maximum waiting time in minutes with the network bandwidth. We can see that our ALB scheme has the shortest waiting time. For the same waiting time requirement, the ALB scheme needs the least bandwidth. For example, suppose there is a video of length $D = 100$ minutes. If the maximum waiting time must be within 1 minute = $0.01D$, the ALB scheme needs about 6 channels. For the same condition, the Live RFS scheme needs about 7 channels, the Live FB scheme needs about 8 channels, and the Live NPB needs 7 channels.

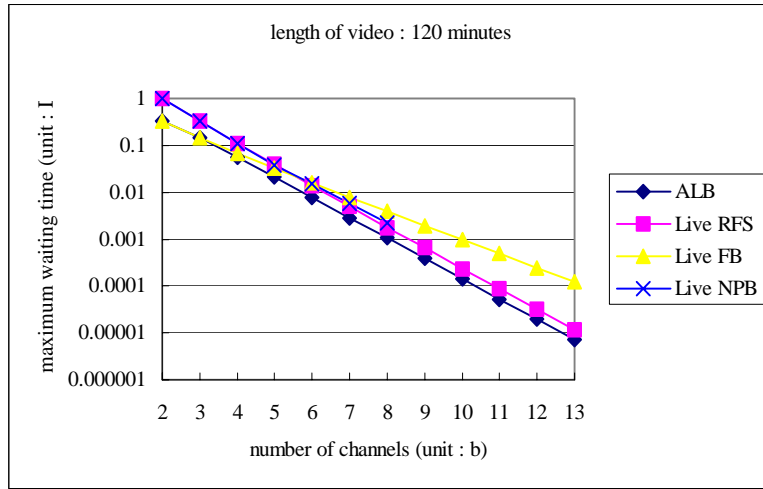


Fig. 19. Maximum waiting time vs. network bandwidth allocation.

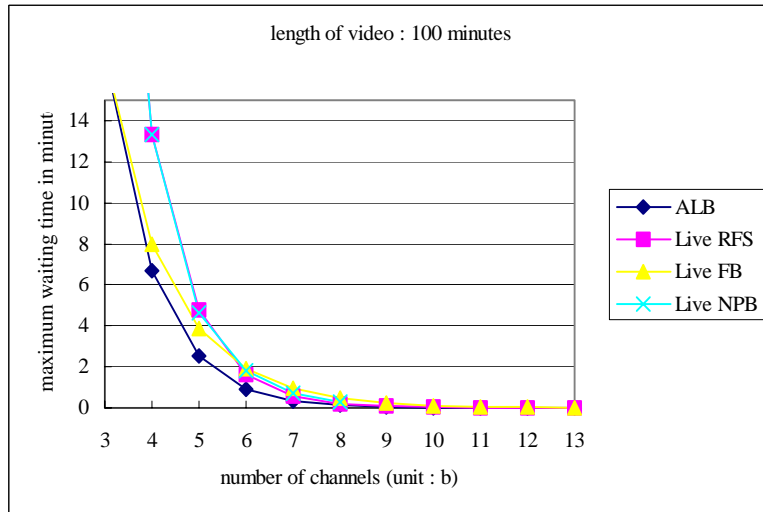


Fig. 20. Maximum waiting time in minutes vs. bandwidth allocation.

4.3 Disk Transfer Rate Requirements at the Client End

At the client end, we write the input video data onto the disk as it needs to be buffered. When we need to consume the data, we need to read the data from the disk. The disk transfer (input/output) rate requirements are the sum of the read requirement and write requirement. According to the client buffer requirements, we find that the maximum disk I/O rate requirements will occur during we read a segment from disk and write the input data from channels $\{C_0, C_2, C_3 \dots C_{\beta-1}\}$ to the disk as shown in Fig. 21. When

the user who enters into the session at 4th time slot needs to playback the segment S_2 , he/she needs to receive and write the input segment S_{11} , S_4 and S_6 from channels $\{C_0, C_2, C_3\}$ simultaneously. Hence, the disk transfer rate requirements of the ALB scheme are β^*b .

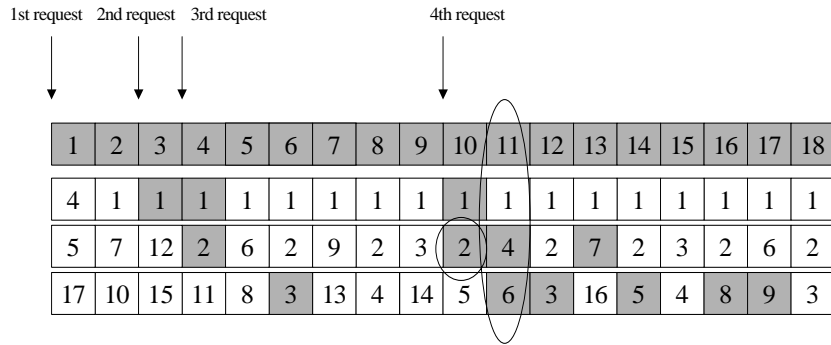


Fig. 21. An example showing the maximum disk I/O rate requirements.

Fig. 22 shows the disk transfer rate requirements for the maximum waiting times of the ALB, Live RFS, Live FB and Live NPB schemes. We can see that our ALB scheme needs the lowest disk transfer rate.

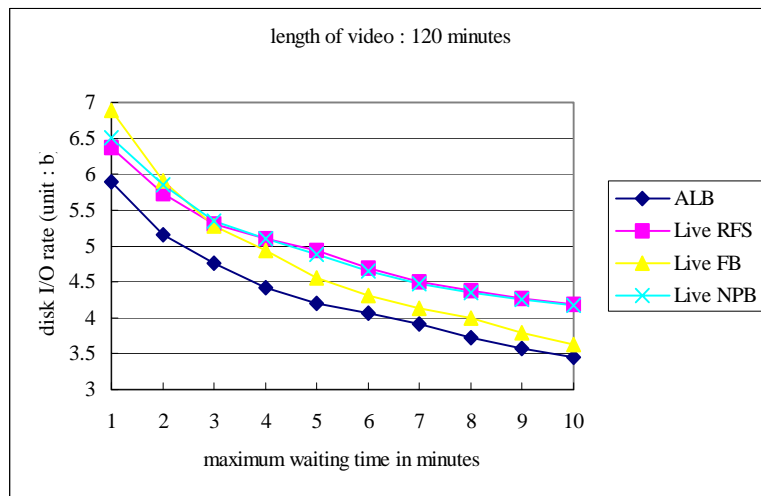


Fig. 22. Disk transfer rate requirements for maximum waiting time.

5. SIMULATION

To evaluate the performance of the ALB scheme, we wrote a simple simulation

program. We assumed that the times of the user requests for a particular video were distributed according to an exponential distribution, $f(x) = \lambda * e^{-\lambda x}$, $x \geq 0, \lambda > 0$. This is because the majority of users will watch the live video on time, and the number of late-comers will decrease with the time. We assumed that a video lasts 127 minutes, which is close to the average duration of a feature video. We partitioned the video into 127 segments, as this would simplify the comparison with UD and AFB. Fig. 23 displays the bandwidth requirements for UD, ST with unlimited extra tapping and an unlimited client buffer, AFB and ALB with a value of λ of 0.14 and from 1 to 100 users. ALB outperformed ST, UD, and AFB when the number of request arrival users was greater than 15. The ST scheme performed slightly better than ALB only when the number of request arrival users was less than 15.

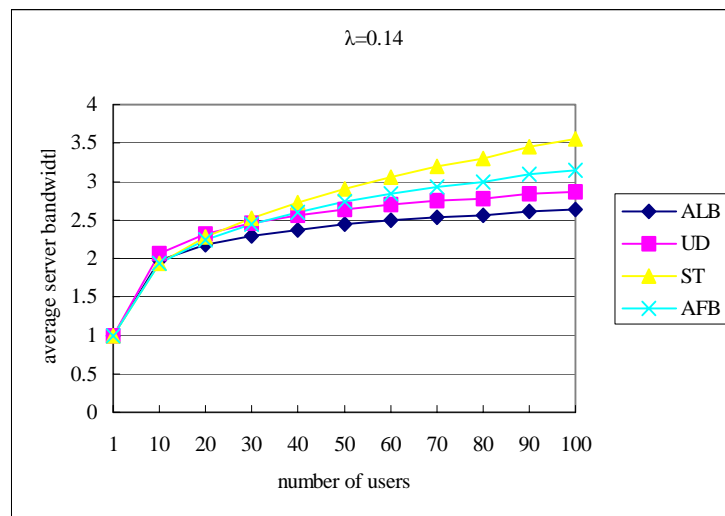


Fig. 23. A bandwidth comparison of ALB, UD, ST, and AFB with a value of λ of 0.14 and from 1 to 100 users.

Figs. 24 and 25 show the bandwidth requirements when the number of users was from 1 to 1000 and from 1 to 10000, respectively. We can find that our ALB scheme still outperformed ST, UD, and AFB.

Fig. 26 depicts the impact of the value of λ on the ALB scheme. The bandwidth requirement for the ALB scheme decreases as the value of λ increases. This phenomenon indicates that requests arrive at the beginning of the live video when the value of λ is larger. Thus, the required bandwidth is less.

6. CONCLUSIONS

With the growth of broadband networks, Video-on-Demand (VoD) has become realistic. One of the important areas worth exploring is how to distribute popular videos

more efficiently. Many significant broadcasting schemes have been proposed to reduce the bandwidth requirements for stored popular videos, but they cannot be used to support live video broadcast perfectly.

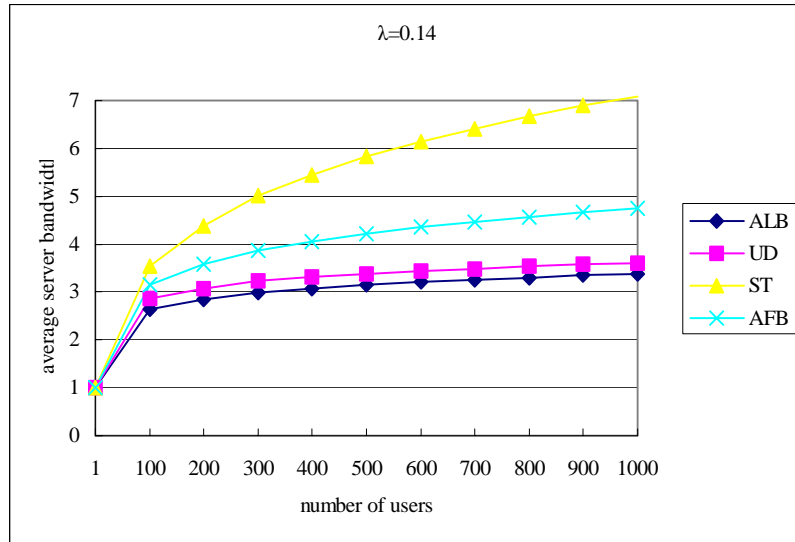


Fig. 24. The bandwidth comparison of ALB, UD, ST and AFB with the value of λ is 0.14 and the number of users is from 1 to 1000.

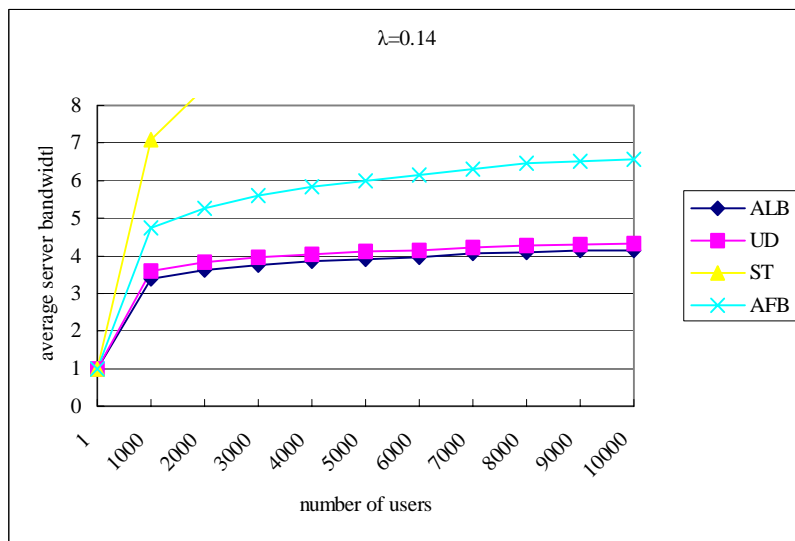


Fig. 25. The bandwidth comparison of ALB, UD, ST and AFB with the value of λ is 0.14 and the number of users is from 1 to 10000.

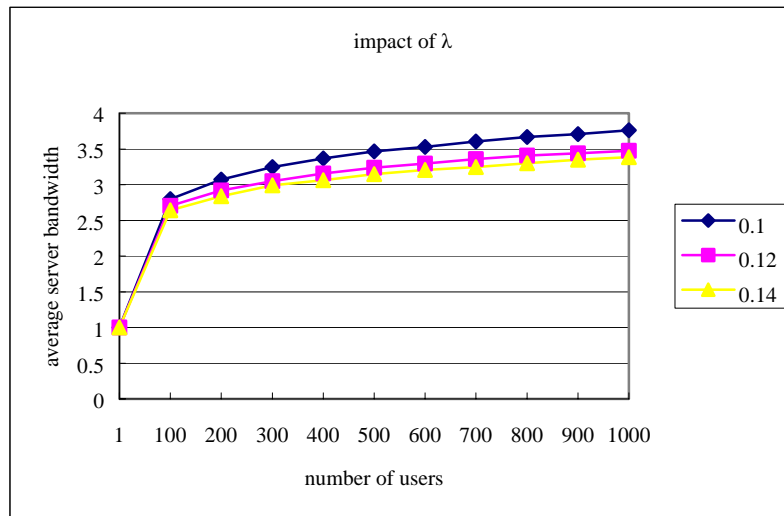


Fig. 26. The impact of λ on the ALB scheme.

Herein, we have proposed a new broadcasting scheme, called the *Adaptive Live Broadcasting (ALB)* scheme, which supports live video broadcasting and performs well over a wide range of request arrival rates. From our analysis and comparison, we find that our ALB scheme is suitable for broadcasting live video. It has several significant advantages: (1) It has the shortest maximum waiting time with fixed channels. (2) It has the least maximum I/O transfer requirements with a fixed maximum waiting time at the client end. A simulation has been employed to evaluate several live broadcasting schemes, such as UD, ST, AFB, and ALB. The results reveal that our ALB scheme consumes the least server bandwidth.

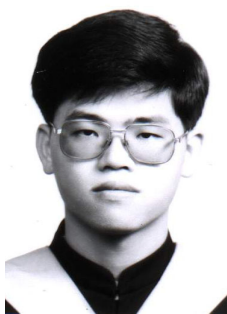
REFERENCES

1. T. Chiueh and C. Lu, "A periodic broadcasting approach to video-on-demand service," *SPIE*, Vol. 2615, 1995, pp. 162-169.
2. S. W. Carter and D. D. E. Long, "Improving video-on-demand server efficiency through stream tapping," in *Proceedings of 5th International Conference on Computer Communications and Networks*, 1997, pp. 200-207.
3. C. H. Chang, J. P. Sheu, and Y. C. Tseng, "A recursive frequency-splitting scheme for broadcasting hot videos in VOD service," M.S. Thesis, Department of Computer Science and Information Engineering, National Central University, Taiwan, 2000.
4. L. S. Juhn and L. M. Tseng, "Fast broadcasting for hot video access," *RTCSA'97: the Proceedings of the 4th international Workshop on Real-Time Computing Systems and Applications*, 1997, pp. 237-243.
5. L. S. Juhn and L. M. Tseng, "Harmonic broadcasting for video-on-demand service," *IEEE Transactions on Broadcasting*, Vol. 43, 1997, pp. 268-271.

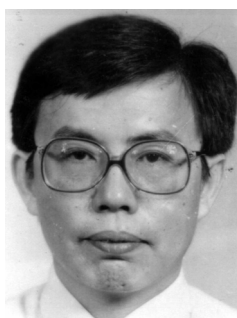
6. L. S. Juhn and L. M. Tseng, "Enhanced harmonic data broadcasting and receiving scheme for popular video service," *IEEE Transactions on Consumer Electronics*, Vol. 44, 1998, pp. 343-346.
7. L. S. Juhn and L. M. Tseng, "Fast data broadcasting and receiving scheme for popular video service," *IEEE Transactions on Broadcasting*, Vol. 44, 1998, pp. 100-105.
8. L. S. Juhn and L. M. Tseng, "Adaptive fast data broadcasting scheme for video-on-demand service," *IEEE Transactions on Broadcasting*, Vol. 44, 1998, pp. 182-185.
9. J. F. Paris, S. W. Carter, and D. D. E. Long, "A hybrid broadcasting protocol for video on demand," in *Proceedings of 1999 Multimedia Computing and Networking Conference*, 1999, pp. 317-326.
10. J. F. Paris, "A simple low bandwidth broadcasting protocol for video on demand," in *Proceedings of 7th International Conference on Computer Communications and Networks*, 1999, pp. 690-697.
11. J. F. Paris, S. W. Carter, and D. D. E. Long, "A universal distribution protocol for video-on-demand," in *Proceedings of 1st International Conference on Multimedia and Expo 2000*, Vol. 1, 2000, pp. 49-52.
12. V. Rangan, H. Vin, and S. Ramanathan, "Designing an on-demand multimedia service," *IEEE Communications Magazine*, Vol. 30, 1992, pp. 56-65.



Hung-Chang Yang (楊宏昌) received the BS degree in mathematics, and the MS degree in computer science from National Central University, Taiwan, in 1998 and 2000, respectively. Currently he is working toward his Ph.D. degree in computer science and information engineering at National Central University, Taiwan. His research interests are in the areas of high-speed networks, IP multicasting, video-on-demand and internet service.



Hsiang-Fu Yu (游象甫) received his B.S. degree in electrical engineering and his MS in computer science from National Central University, Taiwan, in 1993 and 1995, respectively. He currently is working towards his Ph.D. degree in computer science in National Central University. His research interests include proxy cache, telecommunications, information retrieval, and stream delivery.



Li-Ming Tseng (曾黎明) received the BS degree in engineering science and the MS and Ph.D. degrees in electrical engineering from National Cheng Kung University, Taiwan, in 1970, 1974 and 1980, respectively. r. Tseng is currently a professor in the Department of Computer Science and Information Engineering and serves as the head of the Computer Center of National Central University, Taiwan. His research interests are in the areas of distributed systems, computer networks, operating systems, and resource discovery. He is a member of IEEE and the Chinese Computer Association.