Piecewise Patching for Time-shifted TV Over HFC Networks

Wei Xiang, Gang Wu, Qing Ling, and Lei Wang

Abstract — *In video-on-demand (VOD) systems, a number of sophisticated architectures have been proposed to provide instantaneous services through the use of multicast technology. Most of the architectures focus on minimizing bandwidth requirements under the assumption that clients play from beginning to end, but this assumption is not suitable for Time-shifted TV. Moreover, they have been proposed on an IP network, and failed to make full use of the broadcasting property of Hybrid Fiber Coaxial (HFC) network. In this paper, we propose a novel delivery scheme, named as piecewise patching (PP), to tackle these problems. The piecewise patching scheme provides the video data for the users in a piecewise mode instead of a whole patching stream which includes all missing data. The former user can utilize the data from the later users' patching stream whose playback time is in advance of the former user's patching stream. In the proposed scheme, each client receives the data simultaneously from arbitrary channels which broadcast the same content. Simulation results show that the proposed scheme improves the performance of Time-shift TV service significantly in terms of total server bandwidth requirement*[1].

Index Terms — Video Broadcasting, Video-on-Demand, Time-shifted TV, Hybrid Fiber Coaxial.

I. INTRODUCTION

With the rapid development of broadband networking technology and the growth of processor speed and disk capacity, video-on-demand services is one of the most promising services in emerging broadband integrated service digital networks. In recent years, a new video service, Time-shifted TV, becomes popular. Time-shifted TV system provides its subscribers with a virtual VCR (Video Cassette Recording) control to pause, start, stop, rewind and fast forward Live TV programs. Users also have the flexibility to watch any previously broadcasted programs without pre-recording. Different from video-on-demand (VOD), the user starts to play at arbitrary time within the service time of Time-shifted TV. Time-shifted TV can be regarded essentially as VOD service with high levels of interactivity.

In traditional unicast VOD systems, the video server has to reserve a dedicated video channel for each user for the entire duration of the video session. Consequently, the required server and network bandwidth resources increase linearly with the number of concurrent users to support. The bandwidth, especially for the Hybrid Fiber Coaxial (HFC) network, is a very scarce resource and cannot be allocated lavishly. In the HFC network, a single cable is shared by many users. When used for television broadcasting, this sharing scheme does not have adverse impact. All the programs are broadcast on the cable and it does not matter whether there are 10 viewers or 10,000 viewers. When the same cable is used for VOD, it will be a significant difference for 10 users or 10,000 users. If one user decides to watch a film, the corresponding bandwidth is taken away from other users. The more users, the more competition for bandwidth.

To tackle this problem, there has been much work in recent years for reducing the bandwidth requirements of VOD services, by use of multicast delivery techniques. Typical approaches include batching [1-3], periodic broadcasting [4-10], patching/stream merging [11-16], and piggybacking [17]. Different from unicast transmission, data transmitted using multicast enables a server to send a few streams of video data for a large number of clients, thereby significantly reduces the bandwidth requirement of the network. One challenge is to achieve short startup latency under the multicast transmission. Three of these recent techniques, namely, dynamic skyscraper [8], patching/controlled multicast [11-13], and hierarchical multicast stream merging (HMSM) [14, 15], have the key property that they can provide immediate service to each client.

Another challenge is to support interactive playback controls under the multicast transmission. The three algorithms designed for VOD under the assumption that clients play from beginning to end, but this assumption is not suitable for the interactive playback controls. Among the above mentioned three algorithms, dynamic skyscraper does not support VCR operations. Reference [18] shows that, the performance of the other two algorithms modified to support interactive playback control will degrade significantly even at relatively low levels of interactivity. Moreover, the previous patching schemes assume the receive-two model (a client receives two streams simultaneously). Adopting the receive-two model is based on the network bandwidth limitations. Over general IP network, the client, who receives data from several video streams,

[1] This work was supported by Program for New Century Excellent Talents in University (NCET-04-0564) .

Wei Xiang is with the Joint Lab of Network Communication System & Control, Department of Automation, University of Science and Technology of China, Hefei, China (e-mail: xiangwei@mail.ustc.edu.cn).

Gang Wu is with the Joint Lab of Network Communication System & Control, Department of Automation, University of Science and Technology of China, Hefei, China (e-mail: wug@ustc.edu.cn).

Qing Ling is with the Department of Electrical and Computer Enginnering, Michigan Technological University, Houghton, Michigan, USA (e-mail: qling@mtu.edu).

Lei Wang is with the Joint Lab of Network Communication System & Control, Department of Automation, University of Science and Technology of China, Hefei, China (e-mail: wangl@ustc.edu.cn).

Contributed Paper
Manuscript received June 2, 2007

means that the client needs more bandwidth. But for HFC network, the sharing scheme enables the user does not need additional bandwidth for receiving data from several video streams. Namely, the receive-two model can not take full advantage of the HFC network broadcasting properties. Thus, the three algorithms are not suitable for Time-shifted TV over HFC networks.

This paper addresses this challenge over HFC networks by making use of broadcasting property to support Time-shifted TV service. We propose a novel delivery scheme, called Piecewise Patching (PP). Instead of a whole patching stream which delivers the missing data, the Piecewise Patching algorithm provides the missing data in the form of piecewise patching. The length of patching stream is less than a threshold value. When the user receives the data from the patching stream, it also receives the useful data from other streams at the same time. If and only if there is no data to guarantee the continuity of the playback, the user requests a new patching stream. Because of the Time-shifted TV features, the later user may need the same video data in advance of the former user. The server initiates a patching stream for the later user, where the former user also receives the data from the same channel. Consequently, the former user no longer requests a patching stream for these data. The scheme provides the video data in a piecewise mechanism so as to shorten the length of the patching streams for individual users as possible.

The rest of this paper is organized as follows. Section II reviews some related works. The piecewise patching algorithm is proposed in Section III. In Section IV, we analyze the interactivity of the piecewise patching scheme and discuss the optimal threshold. Section V compares the bandwidth requirements between optimized patching, HMSM, and Piecewise Patching through simulations. Section VI states the conclusion of this paper and the direction in the future. Further discussion is facilitated by defining the following notation.

T	service time
N	average number of requests during service time
λ	request rate
W	threshold of the patching stream
B_w	required server bandwidth, in units of the file play rate
t_u	request time of the user.
t_v	playback time of the video.
R	viewing rate, also called consumption.

The request time is that time instant when a viewer/user begins viewing the video. The playback time is the location which a viewer/user requests to play in the video. In general, the playback time is not equal to the request time for Time-shifted TV.

II. RELATED WORK

In this section, we briefly review the related works. Currently, there are three main techniques to provide immediate VOD services, including dynamic skyscraper (with channel stealing)[7,8], patching/controlled multicast[11,13], and hierarchical multicast stream merging (HMSM) [15].

The original skyscraper broadcasting defined in [7] divides a file into K increasing-sized segments with largest segment size. Each segment is continuously broadcast at the file play rate on its own channel. Each client is given a schedule for tuning into each of the K channels to receive each of the file segments.. To achieve lower client request rates, the dynamic skyscraper delivery technique was proposed in [8]. With the technique termed as channel stealing immediate service is provided for clients that are waiting for the start of a segment multicast in a transmission cluster.

Patching proposed in [11] is a dynamic multicast scheme enabling a new request to join an ongoing multicast. The stream patching policy operates as follows. A new service request can exploit an existing multicast by buffering the future stream from the multicast while playing the new start-up flow from the start. Once the new flow has been played back to the skew point, the patching flow can be terminated. The optimized version of this delivery technique is named as controlled multicast [13]. To keep the unicast patch streams short, when the fraction of the file that has been delivered by the most recent multicast exceeds a given threshold, the next client request triggers a new full-file multicast. The optimal threshold is defined as $\min\{(\sqrt{2T\lambda+1}-1)/\lambda, T/2\}$ [13].

Hierarchical multicast stream merging (HMSM) [15] gets inspiration from dynamic skyscraper, patching, and piggybacking. In HMSM, each data transmission stream is multicast. Once two transmission streams are merged, the clients listen to the same stream to receive the remainder of the file. This technique is reasonably close to the minimum achievable required server bandwidth over a wide range of client request rates.

At moderate to high client request rates, the dynamic skyscraper system outperforms the controlled multicast technique, while HMSM outperforms the dynamic skyscraper system for the bandwidth requirements [15]. But the repeating merging process of HMSM leads to high implementation complexity. Meanwhile, the waste of some clients receiving bandwidths usually happens during the merging process.

Dynamic skyscraper is unable to handle the interactivity of time-shifted TV. Therefore we only compare our proposed algorithm with controlled multicast and HMSM in this paper.

III. PIECEWISE PATCHING ALGORITHM

The piecewise patching (PP) scheme is proposed in this section. As described in Section I, the purpose of the scheme is to reduce the bandwidth requirement for Time-shifted TV over HFC network. The algorithm attempts to capture the advantages of stream patching and HMSM technique. In particular, clients are merged using dynamically scheduled patch streams (as in stream patching). Furthermore, each data transmission stream is multicast (as in HMSM).

A. Piecewise Patching Delivery Technique

Key elements of the piecewise patching scheme include:

1. Each data transmission stream is multicast so that any client can listen to the stream. The broadcasting property HFC network guarantees the feasibility of the algorithm.
2. A multi-level patching stream is introduced to reduce the channel redundancy of the conventional patching scheme.
3. The server delivers the data missed in a piecewise mode instead of patching the whole missing data. If and only if there is no required data in existing stream and buffer, the server initiates a finite stream for the client.
4. Once the server provides a patching stream for the later user which needs the video data earlier than the former user, the former user listens to the same stream to receive the remainder.
5. On the server, the information of existing streams is periodically broadcast using Electronic Program Guide (EPG). The client can easily find the useful streams from EPG, and receive the video data from the appropriate streams.

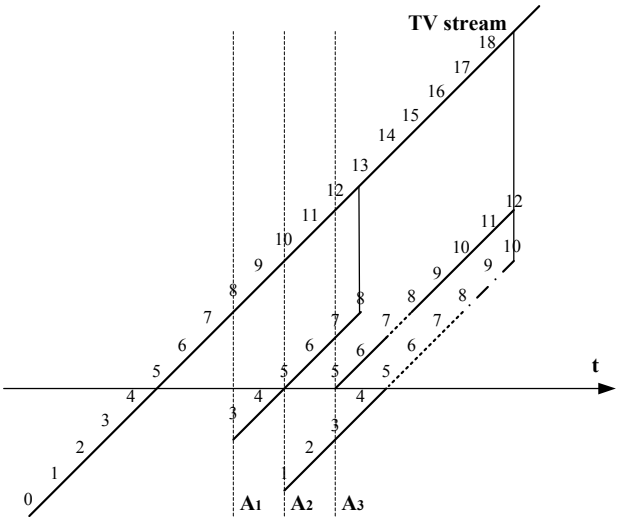

Fig. 1. Merge tree for patching and caching.

For the ease of presentation, we adopt a discrete time model. Without loss of generality, we assume that the video content id divided into segments of unit length and time is also slotted into intervals of the same unit length. Fig.1 shows a simple example of the piecewise patching algorithm. The x-axis is the TV program's play time; the y-axis is the position in TV stream. TV stream is a full stream of the Live TV program. The solid lines represent data transmission streams, which always progress through the file at the play rate. The dotted lines show the amount of useful data that a client has received from the former user's patching stream. The dash dotted lines show the amount of useful data that a client has received from the later user's patching stream.

In the figure, requests arrive from clients A_1, A_2, and A_3 at times 8, 10, and 12, respectively. In order to provide immediate service, each new client is provided a new multicast stream that initiates delivery of the finite portion starting from the requested time point. Client A_1 listens to the TV stream, accumulates data in the client buffer at a rate of R bps, and merges with the TV stream at time 13. Client A_2 listens to the TV stream, receives segments 5, 6, and 7 from stream A_1 during time slots 10, 11, and 13, and receives segments 8 and 9 from stream A_3 during time slots 15 and 16. Client A_3 listens to the TV stream, receives segments 7 from stream A_1 during time slots 12, and requests a new multicast stream at time 15 because of the lack of segments 8, 9, 10, and 11. Consequently, the stream A_3 is discontinuous and composed of the two continuous streams.

B. Client Receiving Schedule

EPG can provide the client receiving schedule, which determines which channel to join based on what video data to receive. Simply speaking, if the buffer space is adequate, the client C can receive video data simultaneously from all useful channels. The client arrives the system at time t_u, and requests the start time t_v. Let (S_0, S_1, …, S_k) be the streams which are served concurrently in the order of the ongoing time, where S0 is the TV stream and S_k is the patching stream for the client C. Furthermore, assume $\tau_0, \tau_1, \cdots, \tau_k$ are the ongoing time of corresponding streams S_0, S_1, …, S_k.

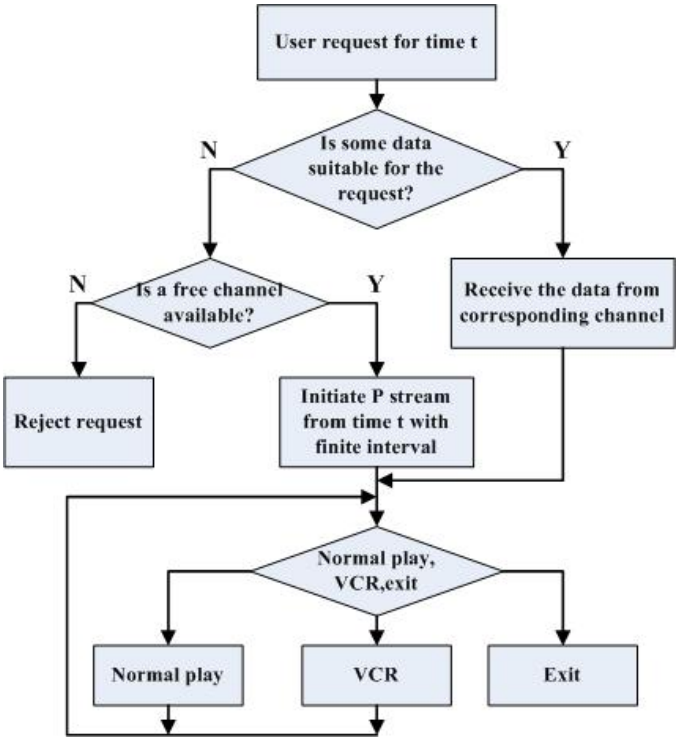

Fig. 2. Flowchart of the play operation.

Fig.2 shows the flowchart of the play operation. First the client retrieves the eligible channel, which can supply instantaneous video-on-demand service within the tolerant time from EPG. If there is not any eligible channel, the client will request the data from the server. If the server is not busy,

it will initiate a patching stream for the user. The client receiving schedule can be described as follows.

Step 1: If $\exists i \in \{0,1,\cdots,k-1\}$ satisfies $\tau_i - t_v < W$, chooses $\min\{\tau_i - t_v\}$ as the patching stream length; else the patching stream length is W.

Step 2: The client starts playback using video data received from the patching stream and caching data from other useful streams which satisfy $t_v < \tau_i$.

Step 3: After the end of the patching stream, the client continues playback with the video data received from the other streams.

Step 4: When the user is in the state of normal play, the client receives periodically the EPG to determine suitable channels which can provide the required video data. Then the client receives the data from corresponding channels, and stores video data in the cache.

Step 5: When there is no data to guarantee continuous playback in the cache, returns to step 1. In order to ensure the continuity of playback, the client usually requests the new patching stream for a period of time in advance.

It is worth noting that the server handles the request similar to the general patching algorithm. Once a multicast stream is initiated, the stream is not affected by the departure of the client terminal doesn't affect the stream continued existence. In other words, the life cycle of the multicast stream has been decided when the stream is created. Each client handles the merge operation by itself. The merge operation doesn't affect the ongoing patching stream. However, the repeating merging process of HMSM results that the patching stream will be adjusted frequently so as to implementation complexity. Therefore, the PP scheme is easier to implement compared with HMSM.

IV. DISCUSSION

A. Interactive Operation

PP scheme enables users to interact with video programs via VCR (video cassette recorder) functions including play, stop, pause, resume, jump-forward, and jump-backward. Several typical interactive operations are described as follows:

Play/Resume: Regular playback from the beginning or any other location.

Stop/pause: Stopping the presentation, without picture or sound (Stop), or with picture but without sound (Pause).

Jump-forward/backward: Immediate jumping to a particular video location in the forward (backward) direction.

Slow Motion: Moving presentation forward slowly, with picture and, possibly, sound.

We present the Piecewise patching support for these VCR functions, when the server simple sources video materials in playback mode. For the interactive requests, the client is given a new stream at the start or end of the interactive request and the new stream is regarded as the general patching stream at the end of the interactive operation. For example, with fast forward, the client may be given a new multicast stream during the fast forward operation and, when the fast forward operation is complete, the new stream is operated as the general patching stream in the standard PP policy.

Fig.3 shows the flowchart for the jump-forward, jump-backward operation. The following discussion focuses on jump-forward, considering the similarity with jump-backward. At time t_0 the user issues a jump-forward operation. Suppose the user jumps to a point in the video t_v seconds in the future. First the client determines whether a jump-forward request for a video location is outside of the buffer. If the requested data is in the buffer, the client consumes the data from the buffer and goes back to normal play. In case of the video location being outside of the buffer, the client will access to information of the ongoing streams in order to determine whether there is an eligible stream in tolerable interval. If the eligible stream is existent, the user resumes normal play at the play point of the eligible stream. Otherwise, the client requests a new patching stream with the interval $\min\{\tau_i - t_v, W\}$, where i=0, 1,…, k-1. If no free channel is available, the previous normal play continues and the operation is failed. Otherwise, the server initiates a patching stream for the user and the user goes back to normal play at the new play point. The operation of jump-backward is similar, except the eligible streams will have negative offsets.

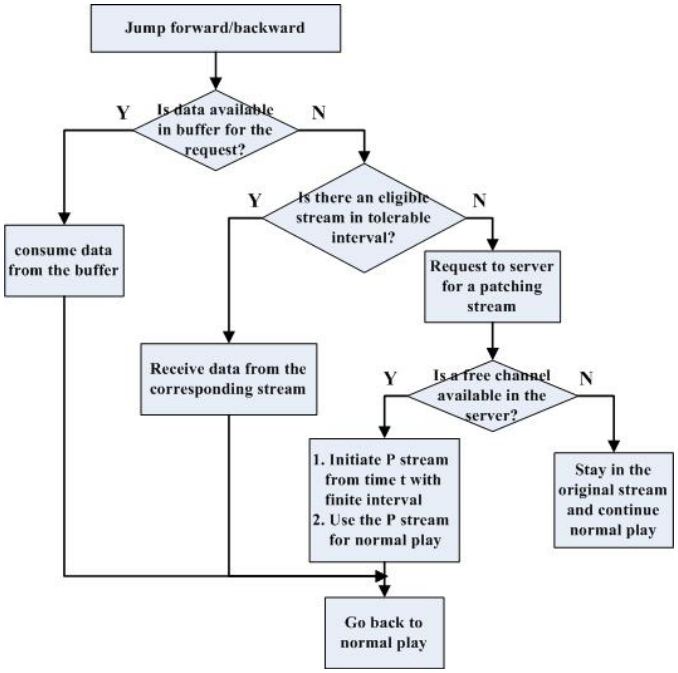

Fig. 3. Flowchart of the jump-forward or backward operation.

In the case of pause and resume operations, in most of the time, the client consumes the video data from the buffer. These operations do not need any extra data request so that there is no impact on the server. Due to the same reason, the slow motion is also regarded as the normal playback for the server.

As described above, the piecewise patching technique is also easily extended for interactive client requests. Moreover,

most of the interactive operations are executed in the client. The server deals with the interaction as a new request. Thus the piecewise patching scheme can be easily realized in the server.

B. Optimal threshold

Recently, Eager et al. [15] proposed the lower bound on the required server bandwidth, for any delivery technique that provides immediate real-time streaming to clients. Consider an infinitesimally small portion of the file that plays at arbitrary time x relative to the beginning of the file. The lower bound on the required server bandwidth measured in units of the play rate, is given by

$$B_{\min imum} = \int_0^T \frac{dx}{x+\frac{1}{\lambda}} = \ln(T\lambda+1) = \ln(N+1) \qquad (1)$$

where N=λT is the average number of requests for the file that arrive during a period of length T.

However, the assumption that clients play from beginning to end is not suitable for Time-shifted TV. Assuming Poisson client request arrivals, the location which a user requests to play in the video obeys uniform distribution. In the best case, the playback time of the patching stream is close to the upper bound of the service time so that other users can share the stream. In the worst case, the playback time of the patching stream is close to the lower bound of the service time so that no user can share the stream. Thus the later user can averagely use the multicast streams created by the former user with the probability of 50% in Time-shifted TV. Approximately, the lower bound on the required server bandwidth is formulized as:

$$B_{\min(Time-shifted\ \mathrm{TV})} = 2\int_0^T \frac{dx}{x+\frac{1}{\lambda}} \qquad (2)$$
$$= 2\ln(T\lambda+1) = 2\ln(N+1)$$

Analytical derivation for the required server bandwidth for piecewise patching appears to be quite difficult to obtain. As an alternative, we discussed the relationship between the required server bandwidth and the threshold of patching stream from numerical results. Fig.4 shows the server bandwidth requirements with respect to 100 to 1000 request rates. The results indicate that for Poisson arrivals and immediate real-time service to each client, the required server bandwidth grows logarithmically with the client request rate. Fig.4 also shows how the server bandwidth requirements, in units of the file play rate, changes as the piecewise patching threshold increases. For example, at the request users of 1,000 during the simulation time, the server bandwidth increases from 23 to 46, while the piecewise patching threshold varies from 1 to 900 second. The trend of increase is not strict, because the random playback request may result that some video data is transmitted consecutively in different channels. Generally speaking, the smaller the threshold is, the less the server bandwidth requested. However, once the threshold exceeds some threshold, the required server bandwidth will converge to a steady value.

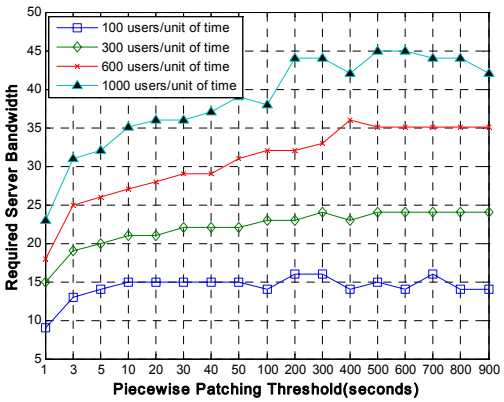

Fig. 4. Required server bandwidth versus piecewise patching threshold. Request rates are 100, 300, 600, and 1000 users per unit of time, respectively.

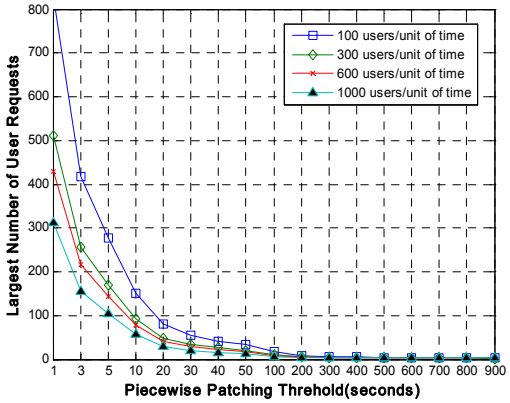

Fig. 5. Largest number of user requests versus piecewise patching threshold. Request rates are 100, 300, 600, and 1000 users per unit of time, respectively.

Small piecewise patching threshold helps to mitigate the server bandwidth requirement, but leads to frequent requests, thus increases system complexity. Fig.5 shows that how the largest number of user requests changes as the piecewise patching threshold increases. When the piecewise patching threshold is 1 second, the number of requests is tremendous. Frequently generating new stream causes serious burden on the system for both the server and the client. As Fig.5 shows, the number of requests decreases exponentially with the increase of the piecewise patching threshold. When the threshold is greater than 200 seconds, the number of requests is acceptable. Comparing Fig.4 and Fig.5, we are able to get a picture of the optimal threshold, as the tradeoff between the required bandwidth and the number of requests.

V. EXPERIMENTAL RESULTS

In this section, we conduct experiments to compare the performance of three algorithms providing immediate service, including optimized patching, HMSM, and Piecewise

Patching. We selected the mean required service bandwidth to support on-demand services as the performance metric. The mean service bandwidth is defined as a function of the client request rate when the server bandwidth is unlimited.

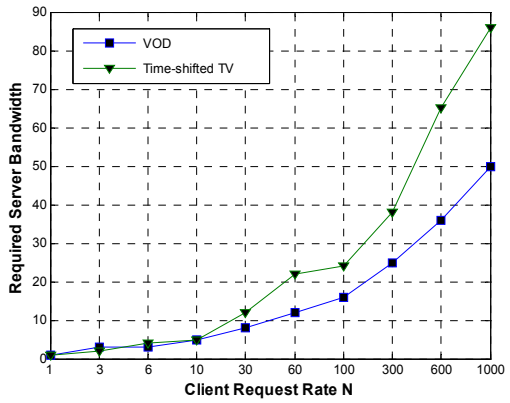

Fig. 6. Required server bandwidth for optimized patching.

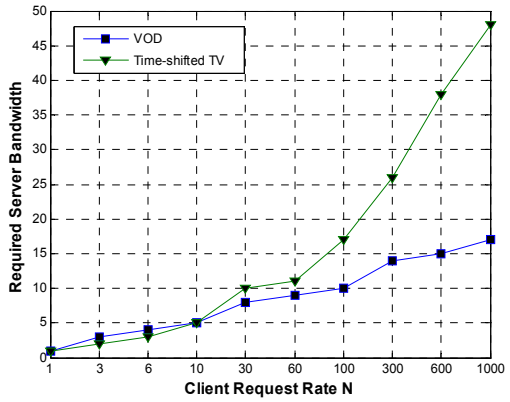

Fig. 7. Required server bandwidth for HMSM.

First, we provide performance comparisons for conventional VOD with Time-shifted TV using discrete-event simulations for optimized patching and HMSM. The Y-axis values are the mean required server bandwidth. The client request rate N is expressed as the number of requests per unit of time. Here the time is defined as the playback time for the file or the service time for Time-shifted TV. Fig.6 and Fig.7 show that as the random access operations of Time-shifted TV, the performance of these multicast streaming algorithms degrades greatly. In the worst case, all subsequent users request the playback point ahead of the previous users, and the system must always create a new stream for the new user which can not merge with an earlier client or group. Under this condition, the optimized patching and HMSM turn into traditional batching techniques. All the multicast streams are regular stream, and each stream forms a group by itself. Fig.8 shows the performance comparison of the worst case. The performances of the optimized patching and HMSM degrade greatly such that the required server bandwidth is close to unicast. Whereas the proposed piecewise patching algorithm introduces the mechanism of piecewise patching, and let the former user utilize the data from the later users' patching stream. In other words, only the data between the request playback time of the later user and the request playback time of the former user is separately served for the former user. The other data receives from the patching streams of the later users.

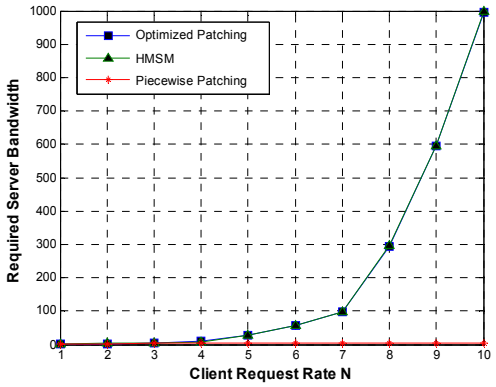

Fig. 8. Required server bandwidth in the worst case.

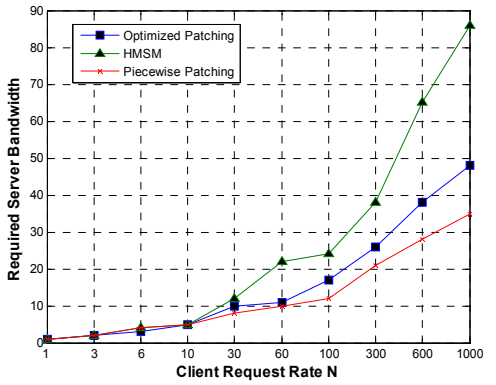

Fig. 9. Required server bandwidth for immediate real-time file delivery.

As shown in Fig.9, the proposed piecewise patching scheme still outperforms the optimized patching and HMSM at high request rate under normal circumstances, in which the users request the random playback points. In contrary to the receive-two mode of HMSM, the multi-receive mode makes full use of the ongoing streams. Furthermore, the piecewise patching scheme allows the former user to make use of the later user's patching stream. So the required bandwidth per client reduces obviously.

VI. CONCLUSION

This paper has proposed a novel patching scheme, piecewise patching (PP), for Time-shifted TV over HFC networks. The piecewise patching scheme provides the video data for the users in a piecewise mode instead of a whole patching stream which includes all missing data. The users can receive the video data simultaneously from multi channels. The former user can utilize the data from the later users' patching stream whose playback time is in advance of the former user's patching stream. Comparing with the optimized patching and HMSM, the piecewise patching has the advantages as follows:

1. The proposed scheme is the first attempt to build the multi-level model for patching, aiming to implement further channel sharing over HFC networks.

2. The piecewise patching technique is suitable for Time-shifted TV. The interactive operations are supported without the degradation of the performance.
3. As shown in the simulation results show that piecewise patching scheme outperforms the previously proposed optimized patching techniques and HMSM, with respect to server bandwidth required for immediate service.
4. The proposed scheme does not bring any additional computation overhead for the server, when comparing to the conventional patching technique.

As the further directions, we will explore the impact of limited buffer and limited channels. Furthermore, developing optimized caching models and strategies will be an important topic.

REFERENCES

[1] A. Dan, D. Sitaram, and P. Shahabuddin, "Scheduling policies for an on-demand video server with batching," in Proc.2nd ACM Multimedia, San Francisco, 1994, pp.15-23.

[2] C. C. Aggarwal, J. L. Wolf, and P. S. Yu, "On optimal batching policies for video-on-demand storage servers," in Proc. 3rd Int. Conf. Multimedia Computing and Systems. Hiroshima, Japan, Jun. 1996, pp.253-258.

[3] S. Sheu and K. A. Hua, "Virtual batching: A new scheduling technique for video-on-demand servers," in Proc.5th DASFAA, Melbourne, Australia, Apr.1997, pp.481-490.

[4] S. Viswanathan and T. Imielinski, "Metropolitan area video-on-demand service using pyramid broadcasting," Multimedia Systems, 1996, vol. 4, no. 3, pp. 197-208.

[5] L. Gao, J. Kurose, and D. Towsley, "Efficient Schemes for Broadcasting Popular Videos", Multimedia Systems, vol. 8, no. 4, pp. 284-294, 2002.

[6] L. Juhn and L. Tseng, "Fast data broadcasting and receiving scheme for popular video service," IEEE Transactions on Broadcasting, 1998, vol. 44, no. 1, pp. 100-105, Mar.

[7] K. A. Hua and S. Sheu, "Skyscraper Broadcasting: A New Broadcasting Scheme for Metropolitan Video-on-Demand Systems," in Proc. ACM Conf. Applications, technologies, architectures, and protocols for computer communication (SIGCOMM '97), Cannes, France, Sep. 1997, pp.89-100.

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

[9] S. Chand and H. Om, "Generalized conservative staircase data broadcasting protocol for video-on-demand," IEEE Transactions on Consumer Electronics, 2006, vol. 52, no. 2, pp. 363-370.

[10] H. F. Yu, H .C. Yang, and L. M. Tseng, "Reverse Fast Broadcasting (RFB) for Video-on-Demand Applications," IEEE Transactions on Broadcasting, 2007, vol. 53, no. 1, pp. 103-111.

[11] K. A. Hua, Y. Cai, and S. Sheu, "Patching: a multicast technique for true video-on-demand services, "in Proc. 6th ACM Int. Conf. Multimedia., Bristol, U.K, Sep. 1998, pp. 191-200.

[12] Y. Cai, K. A. Hua, and K. Vu, "Optimizing Patching Performance," in Proc. SPIE/ACM Conf. Multimedia Computing and Networking 1999 (MMCN'99), San Jose, CA, Jan. 1999, pp. 204-215.

[13] L. Gao and D. Towsley, "Supplying instantaneous video-on-demand services using controlled multicast, "in Proc. IEEE Int. Conf. Multimedia Computing and Systems, 1999, vol. 2, pp. 117-121.

[14] D. L. Eager, M. K. Vernon, and J. Zahorjan, "Optimal and efficient merging schedules for video-on-demand servers," in Proc. 7th ACM Int. Conf. Multimedia, 1999, pp. 199-203.

[15] D. L. Eager, M. K. Vernon, and J. Zahorjan, "Minimizing Bandwidth Requirements for On-Demand Data Delivery," IEEE Transactions on Knowledge and Data Engineering, 2001, vol. 13, no. 5, pp.742-757.

[16] D. L. Guan and S. Y. Yu, "A new patching channel schedule scheme for video multicast," IEEE Transactions on Consumer Electronics, 2003, vol. 49, no. 2, pp. 342-347.

[17] L. Golubchic, J. C. S. Liu, and R. R. Muntz, "Adaptive Piggybacking: a novel technique for data sharing in video-on-demand storage servers, " Multimedia Systems, 1996, vol. 4, no. 3, pp. 140-155.

[18] Y. W. Wong, J. Y. B. Lee, V. O. K. Li, and G. S. H. Chan, "Supporting Interactive Video-on-Demand With Adaptive Multicast Streaming," IEEE Transactions on Circuits and Systems for Video Technology, 2007, vol. 17, no. 2, pp. 129-142.

Wei Xiang received the B.S. and M.S. degrees in automation from the University of Science and Technology of China (USTC), Hefei, China, in 2003 and 2005, respectively. He is currently working toward the Ph.D. degree at the School of Information Science and Technology, USTC. His research interests include peer-to-peer network, video multicast, and video streaming.

Gang Wu received the B.S. and M.S. degrees in automation from the University of Science and Technology of China (USTC), Hefei, China, in 1986 and 1989 respectively. He is currently the Associate Dean of Information Science and Technology at USTC. His research interests include network media services, advanced control and optimization.

Qing Ling (M'07) received the B.S. degree in automation and the Ph.D.degree in control theory and control engineering from the University of Science and Technology of China, Hefei, Anhui, China, in 2001 and 2006, respectively. Since 2006, he has been with the Department of Electrical and Computer Engineering, Michigan Technological University, Houghton, MI, USA. His current research focuses on system modeling, numerical optimization, and signal processing. Dr. Ling received the President Scholarship of the Chinese Academy of Sciences in 2006.

Lei Wang received the B.S., M.S. and Ph.D. degrees in automation from the University of Science and Technology of China (USTC), Hefei, China, in 1994, 1999 and 2004 respectively. His research interests include network media services, complex system modeling and simulation.