A FEEDBACK BASED MULTICASTING PROTOCOL FOR EFFICIENT VIDEO ON DEMAND

Yeonjoon Chung and Ahmed H. Tewfik

Department of Electrical and Computer Engineering,
University of Minnesota, Minneapolis, MN 55455, USA
ychung,tewfik@ece.umn.edu

ABSTRACT

Video on demand services require video multicasting protocols to provide efficient and reliable performances over various client request rates. While reactive protocols perform very well when video demand is low, well-known reactive protocols can't effectively handle these videos in situations of high demand. In this work we have developed an efficient video multicasting protocol, the feedback based multicasting protocol, that requires lower bandwidth to multicast a video at any access rate. The proposed protocol uses an online stream merging scheme requiring lower merging costs than other previous schemes. We have provided both analysis and simulation to show the performance gain over previous protocols. From an analytical method, we found the expected service bandwidth of the feedback based mulitcasting protocol(FMP) is less than that of previous reactive protocols. Furthermore, The FMP shows similar bandwidth requirement to one of the best existing proactive protocols at high client request rates.

1. INTRODUCTION

Handling on-demand streaming of multimedia requires extremely high bandwidth servers and networks in order to service individual customer requests. This situation has resulted in many protocols aimed at reducing the bandwidth requirements of on-demand streaming services. Despite all their differences, we can summarize most of these proposals into two groups [2]. The first group of proposals follow a proactive approach. Proactive protocols provide the most cost-effective solution for distributing popular videos on demand to many clients. The second group take a reactive approach. They achieve the bandwidth reduction by dynamically aggregating clients that make requests closely spaced in time, so that eventually these clients share the same streams. Although we found that the video selection follows a Zipf-like distribution, the frequency of requests for any video is likely to vary widely with the time of the day. While reactive protocols will perform very well when the video is in low demand, they require service operators to consider stable services of given video servers without overloading from client requests. As a stream merging scheme approaches to optimal merging, the dynamic program scheme requires more stream merging costs.

In this work we present an efficient video stream delivery protocol that requires less server bandwidth and lower on-line stream merging costs. The rest of the paper is structured as follows. Section 2 reviews relevant video multicasting protocols and describes properties of them. Section 3 presents the feedback based multicasting protocol, and introduces its theoretical analysis and the basic protocol algorithm. Also presented are the simulation results and discussion of practical aspects concerning the required bandwidth compared to other video multicasting protocols. Finally, we give our concluding remarks in Section 4.

2. PREVIOUS WORK

Patching[7][4] uses a special buffer on the client's set-top box to tap into streams of data on the video-on demand server originally requested by other clients. A new tapping client is also provided a catch-up unicast stream that delivers the initial data that was delivered in the multicast stream prior to the new client's request. The required server bandwidth for delivery of file i, in units of the media play rate, can be derived for Poisson request arrivals as $BW_{patching} = \sqrt{2N_i + 1} - 1$[4]. where $N_i = \lambda_i T_i$ is the average number of requests for the file that arrive during a period of length T_i. T_i is the duration of file i and λ_i is its average request rate.

The hierarchical stream merging protocol [6] provides scalable on-demand streaming that requires less server bandwidth by using patching and merging schemes. Clients that request the same file repeatedly merge into larger and larger groups, leading to a hierarchical merging structure. The HSM doesn't require the set-top box to receive more than two streams at the same time. The required server bandwidth for a given file under the HSM and Poisson requests arrivals can be approximated by the following for-

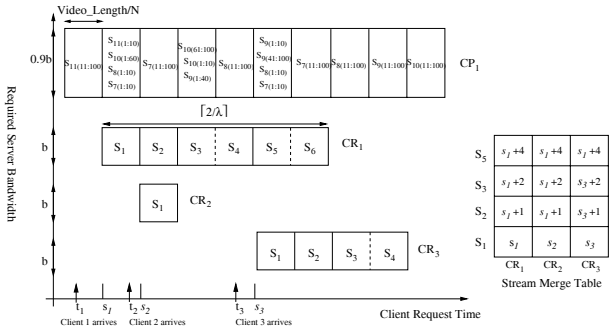

Fig. 1. An overview of the feedback based multicasting scheme when $p = 2$, $\lceil \frac{p}{\lambda} \rceil = 6$, $N = 11$, and $K = 100$.

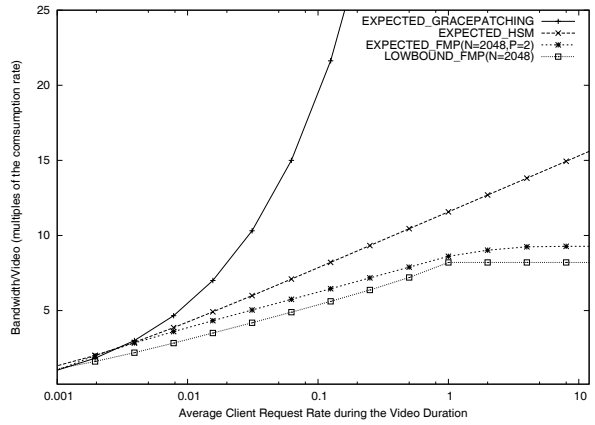

Fig. 2. comparison of the expected bandwidth of the FMP and other protocols

mula [6]: $BW_{HSM} = 1.62ln(N_i/1.62 + 1)$.

Early Merging [5] discussed efficient merge tree schemes using some merging heuristics. Using the dynamic program at each request arrival instant, the optimal merge tree for M currently active streams requires time $O(M^3)$ and space $O(M^2)$[5].

Universal distribution [2] is a reactive approach using a modified fast broadcasting protocol. The video is partitioned into $2^k - 1$ segments of equal duration d. These $2^k - 1$ segments will be grouped into k logical streams. The UD protocol schedules each segment over dedicated streams. For example, if a request arrives during slot i and the last scheduled segment transmission for stream j is before slot $i + 2^{j-1}$, then $i + 2^{j-1}$ becomes the new start slot for stream j. If no transmission of segment S_l of stream j has been already scheduled for any slot greater than i then the server will schedule a new transmission of S_l in slot $b_j + (l - 2^{j-1})$, where b_j is a start slot of stream j.

3. FEEDBACK BASED MULTICASTING PROTOCOL

We propose the Feedback based Multicasting Protocol(FMP) as a video multicasting protocol, which can provide video data at any client request rates with low bandwidth requirement and low stream merging complexity.

The FMP divides a video into N equal segments. The size of the video is $S = L * b$, $S = \sum_{h=1}^{N} S_h$, where L is the duration of the video and b is the consumption rate of the video. Clients wanting to watch a video wait for a small delay interval Δ_i, $0 < \Delta_i \leq \frac{video\_length}{N}$.

The FMP is different from other stream merge schemes in four aspects. First, the FMP selects first $max(2, \lceil \frac{p}{\lambda} \rceil)$ segments of all N segments to transmit over reactive streams, where λ is an expected request rate during a video duration, and p is some integer $p \geq 2$. Second, the FMP maintains dynamic stream merging information onto a small table, the stream merge table, having $O(\lceil \frac{p}{\lambda} \rceil log(\lceil \frac{p}{\lambda} \rceil))$ size, $1 \leq \lceil \frac{p}{\lambda} \rceil \leq N$. Whenever a new client request arrives, the FMP adds the newest stream merge information on the stream merge table. Third, The FMP merges a new request into the stream merge table with $O(log\lceil \frac{p}{\lambda} \rceil)$ merge time costs. Since the FMP looks up only the newest S_1 and S_{2^i+1} in the stream merge table, $0 \leq i \leq \lfloor log_2 \lceil \frac{p}{\lambda} \rceil \rfloor$, when the FMP merges a new request into active streams, the stream merge time costs of the FMP is reduced to $O(log\lceil \frac{p}{\lambda} \rceil)$. Finally, the remaining $N - max(2, \lceil \frac{p}{\lambda} \rceil)$ segments periodically broadcast over a proactive stream. The proactive stream of the FMP follows the basic mapping feature of the SBP [1], which allocates a high frequency segment first on a given bandwidth. A segment S_h of FMP over a proactive stream is subdivided up to K modules, $S_h = \sum_{i=1}^{J} M_i^h$, where J is some integer $1 \leq J \leq K$ and K is some multiple integer of 100. SBP dynamically assigns segments over the given bandwidth while keeping the on-time delivery condition of each module [1]. Unlike SBP, the remaining segments of FMP periodically broadcast over a predefined proactive stream that requires an approximate bandwidth $EBW_{proactive}$, $EBW_{proactive} = \lceil \frac{K}{N-1} * (\sum_{i=max(2,\lceil \frac{p}{\lambda} \rceil)+1}^{N} \lceil \frac{N-1}{i-1} \rceil) \rceil * \frac{b}{K}$. Figure 1 presents an overview of video stream structure of the feedback based multicasting scheme. Since the FMP regularly selects first $max(2, \lceil \frac{2}{\lambda} \rceil)$ segments of a target video to distribute over reactive streams, all remaining segments of a target video broadcast over a proactive stream. First, the scheduler initially makes the stream merge table for S_1, S_2, S_3, and S_5. The FMP adds new merge information onto the table whenever a new request arrives. As shown in Figure 1, client 1 requests a video at t_1. The s_1 represents the service start time of a requested video about a request at t_1. Client 1 can watch a requested video after Δ_1, $s_1 - t_1$, time passed from t_1 over the CR_1 stream and the CP_1 stream, while client 1's

FMP Expected Bandwidth
$min(N, \lceil \frac{1}{\lambda} \rceil) b(1 - e^{-\lambda})(1.5 - 0.5e^{-\lambda}) + \lceil \frac{K}{N-1} \cdot (\sum_{i=max(2, \lceil \frac{2}{\lambda} \rceil)+1}^{N} \lceil \frac{N-1}{i-1} \rceil) \rceil \cdot \frac{b}{K}$

Table 1. Expected Bandwidth of the FMP when the reactive stream size is $max(2, \lceil \frac{2}{\lambda} \rceil)$

set-top box simultaneously stores video segments broadcast over the CP_1 stream. Client 2 arrives at t_2. After looking up a newest column on the stream merge table, the scheduler can immediately merge a requested video stream of client 2 into the newest stream CR_1 at s_2. Client 2 simultaneously listens to CR_1, CR_2, and CP_1. At s_3, client 2 will listen to CR_1 and CP_1. The behavior of client 3 is similar.

As we mentioned before, our FMP uses a hybrid scheme combining a reactive scheme and a proactive protocol. We assume that the requests arrive according to a Poisson distribution with an expected inter-arrival time of $1/\lambda$.

The frequency of multicast of a segment S_i which would be required to watch a movie within a $\frac{video\_length}{N}$ delay requires at least $\frac{1}{i+\lceil \frac{1}{\lambda} \rceil - 1}$, $1 \leq i \leq N$. We can extract a lower bound on bandwidth of the FMP as follows, $LBW_{FMP} = \sum_{i=1}^{N} \frac{1}{i+\lceil \frac{1}{\lambda} \rceil - 1}$. Now recall that the FMP selects first $max(2, \lceil \frac{p}{\lambda} \rceil)$ segments to multicast over reactive streams, where p is some integer $p \geq 2$. We can calculate the expected bandwidth for reactive streams for $p = 2$ as follows. We have simplified client request arrivals in three cases. First, the scheduler receives a client request every $1/\lambda$. The expected required bandwidth of the first case will be $1.5b$ during a $2/\lambda$ duration. Second, the scheduler receives a client request during $2/\lambda$, the expected required bandwidth is b during a $2/\lambda$ duration. In the third case, the scheduler receives no client request during a $2/\lambda$ duration. Since the probability function for the Poisson random distribution is $P[N = k] = \frac{\lambda^k}{k!} e^{-\lambda}$, $k = 0, 1, 2, ..$ We can easily find the required bandwidth for reactive streams($p = 2$) as follows.
$EBW_{reactive} = min(N, \lceil \frac{1}{\lambda} \rceil) b(1 - e^{-\lambda})(1.5 - 0.5e^{-\lambda})$.

Since the FMP periodically broadcasts all remaining segments over a given proactive stream, the required proactive bandwidth is
$EBW_{proactive} = \lceil \frac{K}{N-1} * (\sum_{i=max(2, \lceil \frac{2}{\lambda} \rceil)+1}^{N} \lceil \frac{N-1}{i-1} \rceil) \rceil * \frac{b}{K}$.
The total expected bandwidth of the FMP is then, $EBW_{FMP} = EBW_{reactive} + EBW_{proactive}$, as described in Table 1.

Since λ may be widely varying with time of the day, the FMP may change the size of the reactive streams according to new client request rates. When the scheduler wants to update a reactive multicasting scheme according to the newest client requests, the scheduler can know in advance the required proactive stream bandwidth and the number of segments to broadcast over a proactive stream. Thus, the FMP can provide service consistently to a client, who is already watching a movie, from previous segment mapping information without hard real-time service costs for the new proactive stream.

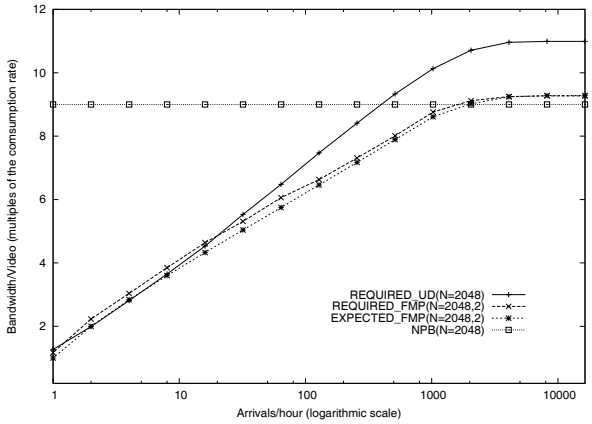

Fig. 3. Bandwidth comparison achieved by the FMP and other protocols about different client request rates

Figure 2 displays the expected bandwidth needed by the FMP with $N = 2048$ and $max(2, \lceil \frac{2}{\lambda} \rceil)$, and other multicast protocols. Note that the Grace patching and the HSM [6] allow instant access to the video while the FMP only guarantees that no client will ever wait more that a $\frac{video\_length}{2048}$ duration, that is no more than 3.5 seconds for a two-hour video. As in Figure 2, the average client request rate on the x-axis are expressed as expected requests of the video duration. The required bandwidth on the y-axis represents multiples of the video consumption rate. As one can see, the bandwidth required by the FMP is definitely lower than that required by the other two protocols throughout x-axis. Further, if the FMP is required to use the lower stream merging costs, the figure shows that the FMP uses a bandwidth that is close to the lower bound when we compared to other protocols.

For simulation, we assume that a video duration of 2 hours and the set-top box has unlimited buffer size. Based on a Poisson distribution with the request rate λ, the simulation ran for 256 hours, which is 128 times of video duration.

Figure 3 displays the average bandwidth needed by the FMP, the UD [2], and the New Pagoda Broadcasting protocol(NPB) [3] when $N = 2048$. When NPB broadcasts 2048 segments over stream channels, total stream channel bandwidth requires more than eight times of the video consumption rate. Since a proactive protocol uses a dedicated bandwidth to broadcast video segments, we select nine times the

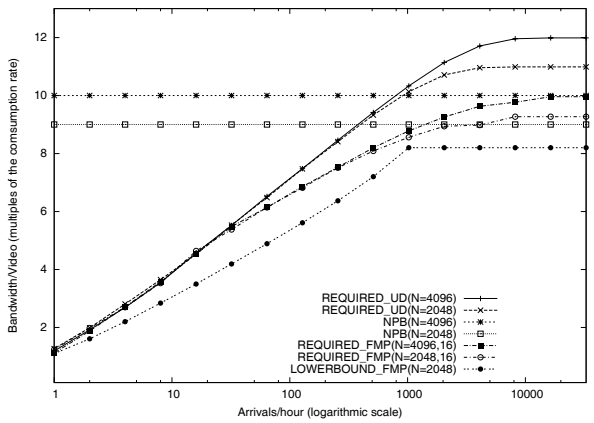

Fig. 4. Bandwidth comparison achieved by the FMP and other protocols with reactive size $max(2, \lceil \frac{16}{\lambda} \rceil)$

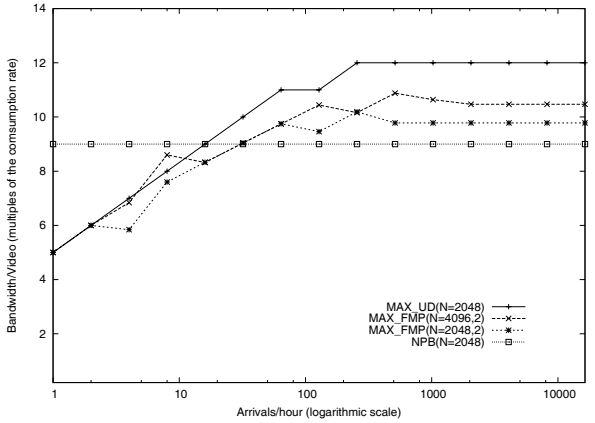

Fig. 5. Compared maximum bandwidth required by the FMP and other protocols during the simulation duration

video consumption rate as a service bandwidth.

Note that FMP, UD, NPB protocol guarantee that clients will watch a video within a $\frac{video\_length}{2048}$ duration, that is no more than 3.5 seconds for a two-hour video. Request arrival rates are expressed in arrivals per hour on the x-axis. The required bandwidth on the y-axis represents multiples of the video consumption rate. As one can see, the bandwidth required by the FMP is lower than that required by the UD protocol as the arrival rate increases. In addition, in high client request scope, where $arrivals > 1000$, the FMP shows similar performance to the NPB protocol [3] that is one of the best proactive protocols. The figure further shows that the simulation based bandwidth requirement of the FMP is very close to its theoretic expected bandwidth throughout x-axis.

Figure 4 compares the average bandwidth required by FMP, UD, and NPB when $N = 4096$ and $N = 2048$ with a reactive size $max(2, \lceil \frac{16}{\lambda} \rceil)$. The numbers on the x-axis display the number of client requests per hour and required service bandwidth presented on y-axis by multiples of the video consumption rate. As shown in Figure 4, the FMP with a reactive size $max(2, \lceil \frac{16}{\lambda} \rceil)$ provides the required bandwidth that is lower than that of UD throughout x-axis. Figure 5 compares the maximum bandwidth requirements of the FMP with those of the UD protocol and the NPB protocol. The numbers on the x-axis display client request arrivals per hour. All quantities on the y-axis represent the maximum bandwidth requirement during 128 times of video duration. The bandwidth of the y-axis represents multiples of the video consumption rate. As one can see, the FMP outperforms the UD protocol as the arrival rate on x-axis increases.

4. CONCLUSION

Although we found that the video selection follows a Zipf-like distribution, the frequency of requests for any video is likely to vary widely with the time of the day. While reactive protocols will perform very well when the video is in low demand, previous reactive protocols can't effectively handle these videos whose requests are frequently in high demand. In this work we have developed an efficient video multicasting protocol, the feedback based multicasting protocol, that reduces the bandwidth required to multicast a video at any access rate. The proposed protocol uses a on-line stream merging scheme requiring low merging time $O(log\lceil \frac{p}{\lambda} \rceil)$. We have provided both analysis and simulation of the performance gain over previous protocols. From a analytical method, We provided the theoretic lower bound throughout client request rates. Furthermore, we provided that our analytical analysis is very close to simulation results about all client requests and our protocol shows similar average bandwidth requirements to one of the best existing proactive protocols at high client request rates.

5. REFERENCES

[1] Y. Chung and A. H. Tewfi k *A Scheduled Broadcasting Protocol for Efficient Video on Demand*, Int'l. Conf. on Acoustics, Speech, and Signal Processing, April, 2003

[2] J. F. Paris, S. W. Carter and D. D. E. Long *A Universal Distribution Protocol For Video-on-Demand*, Proc., Int'l. Conf. on Multimedia and Expo 2000, July, 2000

[3] J. F. Paris, *A Simple Low-bandwidth Broadcasting Protocol for Video-on-Demand*, Proc. Int. Conf. Computer Communication and Network, 1999

[4] L. Gao and D. Towsley *Supplying Instantaneous Video-on-Demand Services Using Controlled Multicast*, Proc. 1999 Int'l Conf. On Multimedia Computing and Systems(ICMCS '99), June, 1999

[5] D. L. Eager, M. Vernon and J. Jahorjan *Optimal and Efficient Merging Schedules for Video-on-Demand Servers*, Proc. 1999 ACM Multimedia Conference, Nov., 1999

[6] D. L. Eager, M. Vernon and J. Jahorjan *Minimizing Bandwidth Requirements for On-Demand Data Delivery*, Proc. 5th Workshop on Multimedia Information Systems, Oct., 1999

[7] K. A. Hua, Y. Cai, and S. Sheu *Patching: A multicast technique for true video-on-demand services* , Proc 6th ACM Int'l Multimedia Conference, Sept., 1998