Providing Scalable On-Demand Interactive Video Service By Means of Multicasting and Client Buffering

S.-H. Gary Chan
Department of Computer Science
The Hong Kong University of Science and Technology
Clear Water Bay, Kowloon
Hong Kong

Edward Chang
Department of Electrical and Computer Engineering
University of California
Santa Barbara, CA 93106-9560
USA

Abstract—**True-VOD provides interactive on-demand (i.e., virtually zero start-up delay) video service by allocating each user a dedicated stream. Such streaming technique, however, cannot scale up to accommodate a large number of users. A more scalable solution is to use multicasting and client buffering. In this paper, we propose a client-initiated (client-pull) on-demand scheme in which short unicast streams are used to "merge" users onto the existing multicast streams by means of pre-buffering. The scheme is observed to trade off some bandwidth with lower buffer requirement as compared with a previously proposed scheme. We then propose and analyze a server-initiated (server-push) scheme for on-demand interactive applications. The scheme is shown to offer service level similar to true-VOD with substantially (many times) lower bandwidth, even with high user interactivity (e.g., averaged ten VCR commands/viewing).**

Keywords— **Video-on-demand (VOD), interactive service, multicasting, client buffering, bandwidth requirement**

I. Introduction

On-demand interactive video service for entertainment or education over the Internet or broadband networks is gaining increasing interest recently. Such streaming is often referred to as video-on-demand (VOD). In true-VOD, a video server allocates a dedicated stream to each user so that the user can freely interact with the displaying video by means of VCR controls (such as "jumping" to anywhere in the video, and pause/start). However, such a system becomes expensive when many concurrent users are to be accommodated. A more scalable solution is to serve multiple requests for the same video with a single multicast stream. In this paper, we address the use of multicasting as a cost-effective solution to provide large-scale distributed on-demand interactive video service.

Today, video multicasting has been used in some cable- or satellite-based movie-on-demand channels, in which requests for a movie arriving within a period of time are "grouped" (i.e., batched) together and served with a single multicast stream [1], [2], [3]. Such batching technique, however, cannot offer flexible interactivity and introduces delay to the users.

In fact, on-demand (i.e., users experiencing no delay) service with user interactive capability can be offered in a multicast environment with some (little) cost in client buffer.[1] We show in Fig. 1 a video system in which a number of servers store all the movies of interest to a large pool of distributed users. Each user has a buffer and is able to receive two streams while playing one. The server streams are partitioned into unicast and multicast streams. The movies are multicast with some delay so as to collect multiple users to a stream. In order to provide on-demand service, unicast streams are initially allocated to a user upon its arrival. While receiving its unicast stream, the client buffers an on-going multicast stream. Once the play-point of the movie is available from the cached data in its buffer, the client is said to "merge" into the multicast stream and the unicast stream can then be relinquished. Note that the unicast streams are "transient" and of short duration. User interactive capability can be provided in a similar manner by pre-buffering a multicast stream closest to the point in which the client is to visit.

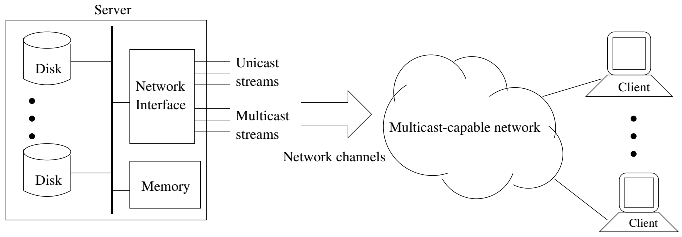

Fig. 1. A video system with multicasting.

Clearly, if the number of the multicast streams is too low (and hence a movie is multicast less frequently), the unicast streams would be held longer, leading to a higher number of required unicast streams. Given the movie request rate, there is hence a trade-off between the number of unicast and multicast streams, and it is of interest to minimize the sum of them, corresponding to the number of concurrent streams (and thereof network channels) required. (Note that such optimization also corresponds to delivering the least amount of data to serve the user pool.)

There are in general two ways to deliver video data to the users. In the *client-initiated* (or "client-pull") approach, the start of a multicast stream is driven by the arrival of the users; while in the *server-initiated* (or "server-push") approach, the server multicasts the movies independent of the arrival of users. In this paper, we first propose a client-initiated scheme for on-demand video service, and analyze it in terms of its stream and buffer requirements. By comparing it to a previously proposed scheme [4], the scheme has a lower buffer requirement with some cost on bandwidth. We then propose a server-initiated scheme for on-demand interactive service. As compared to true-VOD (i.e., allocating one stream per user), our scheme is able to offer similar type of service with substantially (many times) lower bandwidth.

We describe briefly the previous work on client buffering as follows. Offering on-demand service by a technique termed "patching" has been analyzed in [4], [5], [6], [7], [8], [9]. We differ from these reported schemes in terms of how movies are multicast (there is a periodic multicast schedule in our schemes), and consider offering *interactive* capability here. A novel client buffering scheme is proposed in [10], in which the clients, instead of the server, is used to stream videos. Though the scheme leads to substantial saving in server streams, it does not lead to lower network bandwidth requirement. Providing interactivity in the multicast environment has been discussed in [11], [12], [13]. We propose and analyze here a new and simple scheme (in terms of how the streams are merged) to support such a service. There is also a body of work on the use of client buffering for those hot movies to *reduce* user delay to a certain value. (see, for example, [14], [15], [16], [17], [18], [19], [20]). These schemes, however, do not readily extend to on-demand streaming, which we consider there. The use of

This work was supported, in part, by the Hong Kong Telecom Institute of Information Technology and Sino Software Research Institute at the HKUST.

[1] In this paper, we refer to a "user" as a person in the system, while a "client" is the set-top box the user uses.

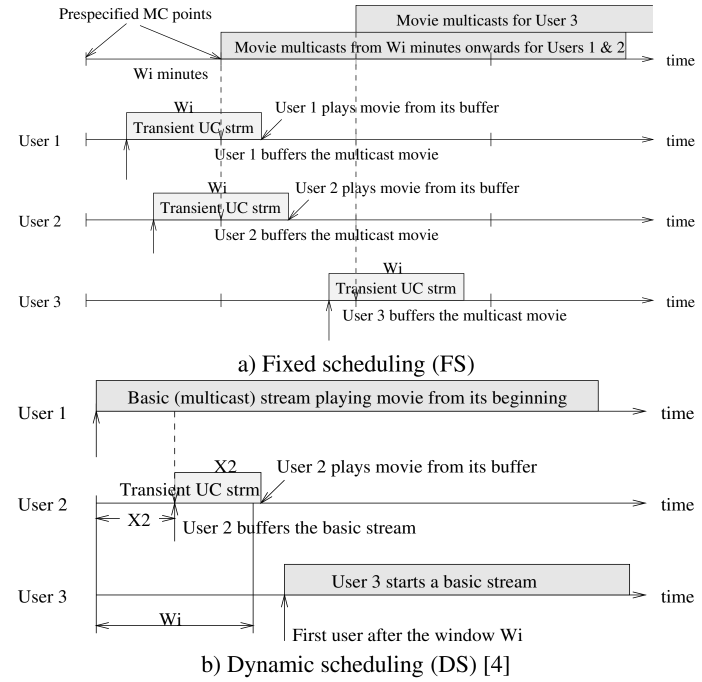

Fig. 2. Schemes to provide on-demand video service with multicasting.

hard disks to economically utilize a client's buffer has been reported in [21], while the use of server caching, rather than client buffering, to achieve cost-effective on-demand video services has been reported in [22], [23].

This paper is organized as follows. In Sect. II, we introduce the client-initiated on-demand scheme and address how server/network bandwidth can be minimized by optimizing the multicast interval given movie's request rate. In Sect. III we present the server-initiated scheme to offer on-demand interactive service, and analyze its stream and buffer requirements. We conclude in Sect. IV.

II. Client-Initiated On-Demand Schemes

In this section, we focus on the schemes to provide on-demand service (without user VCR control besides start/stop). Each client has a buffer large enough to cache B minutes of video (B is about 10–20 minutes). We first present the schemes, followed by deriving the requirements in client buffer size, network channels and server streams, given a certain target (maximum) request rate the server is designed under. We then optimize the system parameters (multicast intervals and server streams) to minimize the number of network channels used.

Let N_v be the number of movies in the system indexed from $i = 1, 2, \ldots, N_v$, and a user requests movie i with probability ν_i (clearly, $\sum_i \nu_i = 1$). The aggregate request rate to the system is Λ req/min; therefore, the request rate for movie i is given by $\lambda_i = \Lambda \nu_i$ req/min.

A. Scheme Descriptions

To offer on-demand service, the total number of server streams (the sum of multicast and unicast streams) should be large enough so that the probability of running out of streams is very low, say, 10^{-3} (we address how this number of streams is determined later in Sect. II-B). Under this condition, the servicing of a movie is independent of the servicing of the other movies, and we can focus on a particular movie i. We study the following two schemes pertaining to how the multicast is scheduled (Ref. Fig. 2):

Fixed scheduling (FS): In this scheme, the movie is multicast at "fixed" regular intervals separated by W_i minutes. For each multicast point, the movie is delivered from segment W_i minutes onwards. Upon the arrival of a user, the initial "start-up" portion of the movie (of W_i minutes) is supplied to the user using a unicast stream. While the start-up portion is being played, the multicast stream of the movie would be started and buffered by the client for its future use. Clearly, all requests arriving between two consecutive multicast points share the same multicast stream. If no user arrives within an interval, the multicast in the following point is skipped (or canceled).

Dynamic Scheduling (DS): This scheme was proposed in [4]. We describe it here for comparison purpose and present a different analytic approach. As opposed to fixed scheduling, the multicast points in this scheme is not fixed but "dynamically" driven by the arrival of requests: the first user after W_i minutes of a multicast starts a new multicast stream (the basic stream) of the movie. An arrival within W_i minutes from the start of the basic stream pre-buffers the multicast data while a transient unicast stream is used to supply the missing (beginning) portion of the movie so as to merge the user back to the basic stream. Clearly, the duration of the unicast stream is equal to the difference between the request arrival and the start of the basic stream.

Note that if W_i is too long, the number of unicast channels required would be high; on the other hand, if W_i is too short, the number of multicast channels required will be high. Therefore, given a certain request rate, there is an optimal W_i so as to minimize the total number of unicast and multicast channels. To formulate the problem, let $\overline{N}_{u,i}$ and $\overline{N}_{m,i}$ be the average number of unicast and multicast channels used, respectively, for movie i in the system, $1 \le i \le N_v$. Let

$$\overline{N}_{c,i} \triangleq \overline{N}_{u,i} + \overline{N}_{m,i} \tag{1}$$

be the total number of channels used for movie i. We would like to choose W_i so as to minimize the total number of channels in the system, i.e.,

$$\min_{W_i} \left(\overline{N}_c \triangleq \sum_{i=1}^{N_v} \overline{N}_{c,i} \right),$$

subject to the buffer requirement $W_i \le B$.

To address this problem, first we observe that we can first consider the unconstraint problem to obtain W_i^*. If $W_i^* \le B$, the constraint on the buffer size is redundant; on the other hand, if $W_i^* > B$, we simply set $W_i^* = B$ (since $N_{c,i}$ is decreasing in W_i when $W_i \le B$). Further observe that since the movies are scheduled independently, we can focus on a particular movie i and simply consider

$$\min_{W_i} \overline{N}_{c,i},$$

as if there is no constraint on W_i.

B. Scheme Analysis

In this section, we analyze the schemes in terms of their requirements in unicast and multicast channels with Poisson arrival. Clearly, we only need to focus on the case with $\lambda_i T_h \ge 1$, otherwise there is no point in using multicasting. Let T_h minutes be the length of the movie.

FS: Since each arrival requires a unicast channel of length W_i minutes, we have

$$\overline{N}_{u,i} = \lambda_i W_i. \tag{2}$$

Because a multicast channel of length $(T_h - W_i)$ minutes is started *if* there is one or more arrivals within a slot (with probability $1 - e^{-\lambda_i W_i}$), by Little's formula, we have

$$\overline{N}_{m,i} = (T_h - W_i)\frac{1 - e^{-\lambda_i W_i}}{W_i}. \tag{3}$$

By minimizing the sum of Eqs. (2) and (3), we obtain W_i^*.

DS: Note that on average there are $1 + \lambda_i W_i$ users sharing a multicast stream (the first term is due to the first user which starts

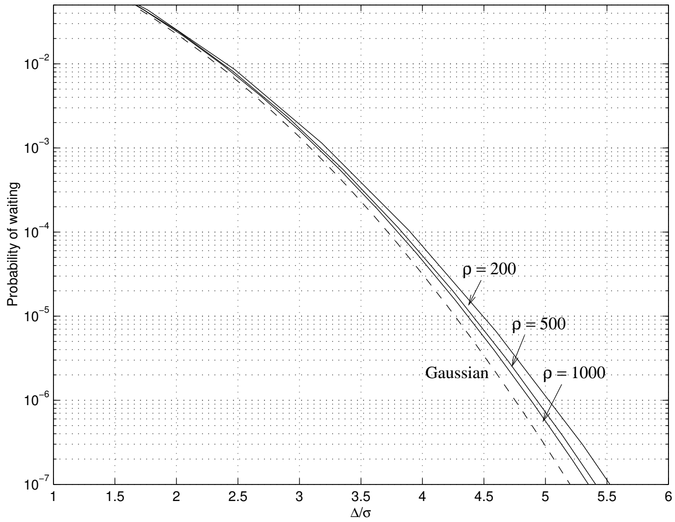

Fig. 3. Probability of running out of streams vs. Δ/σ.

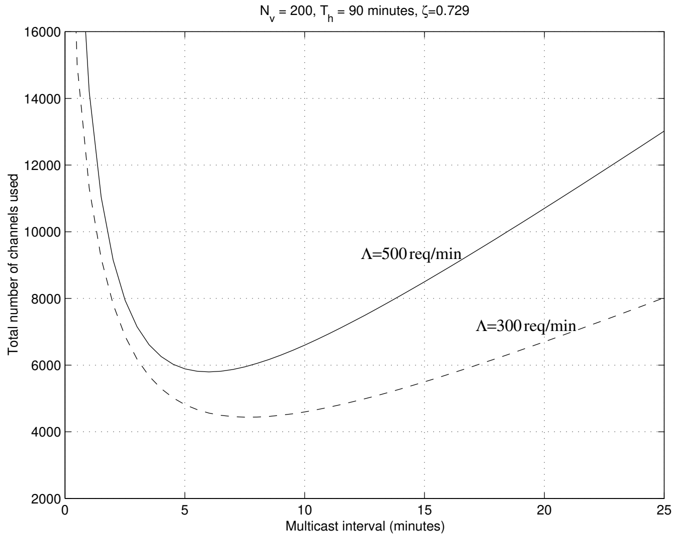

Fig. 4. $\overline{N}_c$ vs. W for fixed scheduling (FS) with uniform W.

the basic stream), out of which $\lambda_i W_i$ users are served with unicast streams. Therefore, the request rate for unicast streams is $\lambda_i W_i/(1+\lambda_i W_i)\lambda_i$. Since each unicast stream is held on average for $W_i/2$ minutes, using Little's formula, we hence obtain

$$\overline{N}_{u,i} = \lambda_i \left(\frac{\lambda_i W}{1+\lambda_i W_i}\right) \frac{W_i}{2}. \quad (4)$$

Furthermore, since the basic (multicast) streams (of length T_h minutes) are started on average every $W_i + 1/\lambda_i$ minutes, we have (using Little's formula again),

$$\overline{N}_{m,i} = \frac{T_h}{W_i + 1/\lambda_i}. \quad (5)$$

Minimizing the sum of Eqs. (4) and (5), we have

$$W_i^* = \frac{\sqrt{1+2\lambda_i T_h} - 1}{\lambda_i}. \quad (6)$$

Given W_i, we now derive the server streams required, M_s so that the probability of running out of such streams is less than some small number ε (e.g., $\varepsilon \approx 10^{-3}$ or lower). Clearly, the number of multicast streams required is $\sum_i \lceil T_h/W_i \rceil$. What remains to be determined is the number of unicast streams. Note that the request rate for the unicast streams is Poisson with rate $\Lambda_U = \sum_i \lambda_i = \Lambda$ for FS, and $\Lambda_U = \sum_i \lambda_i(\lambda_i W_i/(1+\lambda_i W_i))$ for DS. Let the average holding time of the unicast streams be $\overline{W}$ (which is equal to $\sum_i \nu_i W_i$ for FS, and $\sum_i \nu_i W_i/2$ for DS). The use of unicast streams can then be modeled by an $M/G/\infty$ system with arrival rate Λ_U, from which it is well-known that the probability of n unicast streams being in use is given by the Poisson distribution $\pi_n = (\rho^n/n!)e^{-\rho}$ with mean ρ and variance $\sigma^2 = \rho$, where

$$\rho \equiv \Lambda_U \overline{W}. \quad (7)$$

Let N be the random variable corresponding to the number of concurrent streams used. The number of unicast streams required, M_u, is obtained by solving $P(N \geq M_u) = \sum_{n=M_u}^{\infty} \pi_n \leq \varepsilon$. Since $M_u \geq \rho$, we may write $M_u = \rho + \Delta$, $\Delta \geq 0$. We show in Fig. 3 $P(N \geq M_u)$ versus Δ/σ, given ρ. Also shown in dashed line is a Gaussian distribution with the same mean and variance. We see that the Poisson tail is very close to a Gaussian, especially when its mean is high. Therefore, for $\varepsilon \approx 10^{-3}$–$10^{-6}$, we need $M_u = \rho + z\sqrt{\rho}$, where $z \approx$ 3–5. M_s is hence the sum of M_u and $\sum_i \lceil T_h/W_i \rceil$.

C. Illustrative Numerical Examples and Comparisons

In this section, we present some illustrative numerical results on the schemes. As in most previous work, we consider that the movie popularity ν_i is Zipf-distributed, i.e., $\nu_i \propto 1/i^\zeta$ ($\zeta \approx 0.729$ according to some actual rental data [12]). We also consider a baseline movie system with $\Lambda = 500$ req/min, $N_v = 200$, $\zeta = 0.729$, and $T_h = 90$ minutes.

We first consider FS with the same W for all movies (i.e., uniform window size). We show in Fig. 4 $\overline{N}_c$ versus W, given Λ. As expected, when W increases, $\overline{N}_c$ first decreases (as more and more users share a multicast stream) and then increases (due to longer unicast streams). There is an optimal W^* achieving minimum $\overline{N}_c$ (at around 6 minutes for $\Lambda = 500$ req/min). This figure shows that an arbitrarily chosen W may lead to excessive bandwidth requirement. Note that too high a value of W is counter-productive in the sense that it results in excessive client buffer without any saving in bandwidth. We see that if W is chosen appropriately, this "one-size-fits-all" approach leads to a substantial decrease in channel requirement with low client buffer size (around 6,000 channels and 6 minutes of client buffer in this case) as compared to the true-VOD case (which requires $\Lambda T_h = 45,000$ channels). When Λ decreases, W^* increases so as to merge more requests together in a multicast stream.

We next compare the optimal window size of a movie, W_i^*, for FS and DS versus the movie request rate λ_i in Fig. 5. When video popularity increases, the optimal window size decreases so as not to use too many unicast channels. We see that the client buffer does not have to be very large (less than 20% of the movie length in most cases). Our scheme FS requires less client buffer as compared to the proposed DS, by roughly 20-30%.

We show in Fig. 6 the corresponding $\overline{N}_{c,i}^*$ versus λ_i for DS and FS (one with uniform W of 6 minutes and another one with W_i^*), and the true-VOD case. As λ_i increases, $\overline{N}_{c,i}^*$ correspondingly increases. Clearly, the more popular a movie is, the more bandwidth it requires. Note that while FS achieves lower buffer requirement than DS, it has higher bandwidth requirement, especially when the movie is popular. Both schemes lead to substantial saving in bandwidth as compared with the true-VOD case. For FS, the approach with uniform W does not perform as well. This result shows that a knowledge of the arrival process of a movie can lead to substantial saving in the system.

We conclude this section by summarizing in Table I the bandwidth requirement of the schemes for the video system ($\Lambda = 500$ req/min,

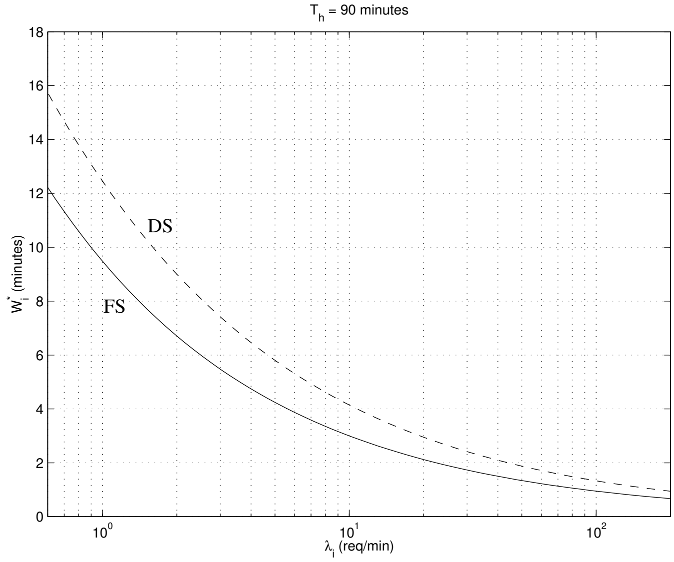

Fig. 5. Comparison of W_i^* (client buffer requirement) vs. λ_i for FS and DS.

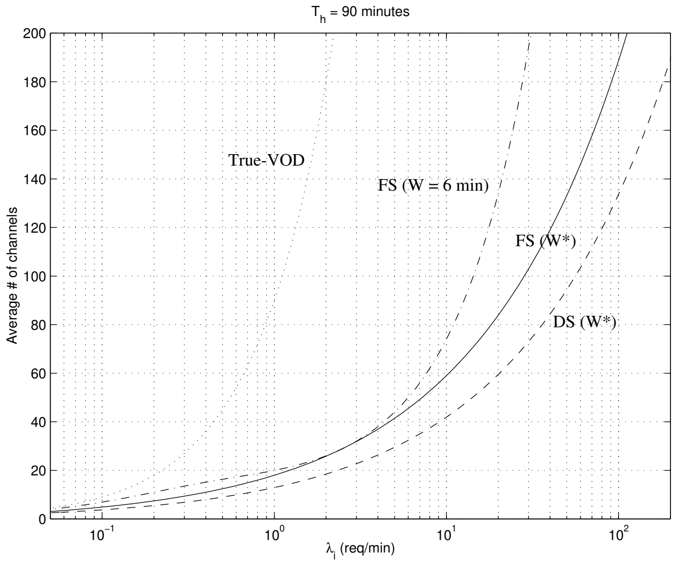

Fig. 6. $\overline{N}_{c,i}^*$ vs. λ_i for different schemes.

$N_v = 200$, $\zeta = 0.729$ and $T_h = 90$ min). Clearly, all the buffering schemes achieve substantially lower channel requirement as compared to true-VOD. Note that even though FS (with W^*) requires about 20-30% lower client buffer than DS, its bandwidth requirement is higher (by about 40%). In the actual system deployment, such trade-off between client buffers and bandwidth requirement has to be carefully considered. As compared to the true-VOD case, the table shows that an order-of-magnitude reduction in bandwidth requirement can be achieved with some sacrifice of user interactivity and a little cost in client buffer.

III. A SERVER-INITIATED ON-DEMAND INTERACTIVE SCHEME

In this section, we propose and study a scheme to offer on-demand interactive service. In this scheme, movie i is multicast in a staggered manner every W_i minutes from its beginning till its end no matter whether there is any arrival within a slot or not. Clearly, if we order the video segments played by the multicast streams at any time, the segments are offset by W_i minutes. In other words, if the first stream

TABLE I

NETWORK CHANNEL REQUIREMENT FOR DIFFERENT SCHEMES FOR ON-DEMAND SERVICE ($\Lambda = 500$ REQ/MIN, $N_v = 200$, $\zeta = 0.729$ AND $T_h = 90$ MINUTES).

Scheme	$\overline{N}_c$
True-VOD	45,000
FS (uniform $W = 15$ min.)	8,500
FS (uniform $W^* = 6$ min.)	5,796.62
FS (with W_i^*)	5,182.4
DS	3,705.98

is playing the segment T minutes from the beginning of the video, then the segment played by the jth stream would be $T + jW_i$ minutes, for $j = 0, 1, 2, \ldots, \lceil T_h/W_i \rceil - 1$. A client visiting a particular play-point of a movie (note that the play-point could be the beginning of the movie) first joins the multicast stream which is delivering a later movie segment which is closest to that play-point, i.e., if the segment the user wants to visit is S minutes, then the index of the streams, j^*, that the user joins would be given by $T + (j^* - 1)W_i \leq S < T + j^* W_i$. Two special cases are that if $S < T$, then $j^* = 0$; and if $S > T + (\lceil T_h/W_i \rceil - 1)W_i$, there is no need to join any multicast stream. The user then pre-buffers the multicast stream, and a unicast stream of length $T + j^* W_i - S$ ($T_h - S$ in the case of $S > T + (\lceil T_h/W_i \rceil - 1)W_i$) is allocated to merge the user back to the multicast stream.

To analyze this scheme, we consider that a user on average executes χ number of interactive commands during the duration of their viewing after its arrival, and hence the request rate of the interactive commands is given by $\lambda_{I,i} = \chi\lambda_i$. We consider the worst case (the pessimistic case) in which each interactive command leads to a jump out of the user's buffered data (and hence leads to an allocation of an unicast stream) and the user is completely merged with the multicast stream before it executes another control command again. Under this circumstance, the unicast streams on average are of length $W_i/2$ minutes. Summing up the multicast and unicast components, we clearly have

$$\overline{N}_{c,i} = \frac{T_h}{W_i} + (\lambda_i + \lambda_{I,i})\frac{W_i}{2}, \tag{8}$$

which yields

$$W_i^* = \sqrt{\frac{2T_h}{\lambda_i + \lambda_{I,i}}}, \tag{9}$$

and hence $\overline{N}_{c,i}^* = \sqrt{2T_h(\lambda_i + \lambda_{I,i})}$. From this, we see that as users interact more often, W_i^* (and hence the client buffer requirement) decreases, while the bandwidth requirement increases.

We show in Fig. 7 the total channels required $\overline{N}_c^*$ with respect to the total request rate Λ for the video system we considered before ($N_v = 200$, $\zeta = 0.729$, and $T_h = 90$ minutes). We compare the bandwidth requirements for true-VOD (given by ΛT_h), the previously proposed scheme DS, and the on-demand interactive scheme proposed in this section. As Λ increases, $\overline{N}_c^*$ increases, with the margin between the scalable schemes and true-VOD widening. Overall, our on-demand interactive scheme offers substantially lower system requirement than true-VOD while offering similar type of service, especially with the request rate is high. Without user interactivity, an on-demand system cuts the cost further.

IV. CONCLUSION

In true video-on-demand, each user is assigned a dedicated stream for the duration of the movie, thus offering on-demand (i.e., no initial user delay) interactive service. Such a system becomes expensive as thousands of concurrent users have to be accommodated. A much more scalable solution is to use multicasting. We have studied in this paper the use of multicasting and client buffering to offer scalable on-demand

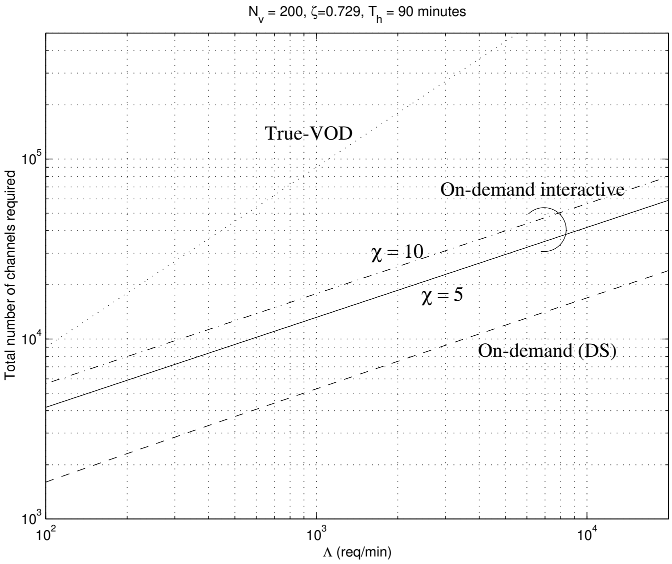

Fig. 7. Channel requirement vs. Λ for different schemes.

interactive service, with the objective to design a low-cost system (with respect to network and server bandwidth requirements) given the external request rate.

We study a client-initiated scheme (Fixed Scheduling) for on-demand service, in which the multicast of a movie is driven by the arrival of requests and unicast streams are used to "merge" the users back to the existing multicast streams. We analyze its performance, and as compared to a previously proposed scheme based on "patching" (Dynamic Scheduling), our scheme achieves a lower buffer requirement with the cost of some bandwidth. Therefore, in the actual design of the system, the decrease in such user buffering has to be traded off with the increase in bandwidth requirement. We also propose and study a server-initiated on-demand interactive system. Our system achieves substantially lower (many times lower) bandwidth requirement as compared with the true-VOD case while offering similar type of service even when the user interactivity is not low (averaged 10 VCR commands per viewing in our example). Our results indicate that multicasting with client buffering is able to offer scalable on-demand interactive service, and a knowledge on the request rate of the movies can lead to substantial cost saving as compared with the case in which such information is not known.

REFERENCES

[1] T. Little and D. Venkatesh, "Prospects for interactive video-on-demand," *IEEE Multimedia Magazine*, pp. 14–24, Fall 1994.

[2] V. O. K. Li and W. Liao, "Distributed multimedia systems," *Proceedings of the IEEE*, vol. 85, pp. 1063–1108, July 1997.

[3] F. A. Tobagi, "Distance learning with digital video," *IEEE Multimedia Magazine*, pp. 90–94, Spring 1995.

[4] L. Goa and D. Towsley, "Supplying instantaneous video-on-demand services using controlled multicast," in *Proceedings of IEEE Multimedia Computing Systems*, June 1999.

[5] K. A. Hua, Y. Cai, and S. Sheu, "Patching: A multicast technique for true video-on-demand services," in *Proceedings of ACM Multimedia 98*, (New York, USA), pp. 191–200, Sept 14-16 1998.

[6] Y. Cai, K. A. Hua, and K. Vu, "Optimizing patching performance," in *Proceedings of Multimedia Computing and Networking (SPIE Vol. 3654)*, (San Jose, CA), pp. 204–15, Jan. 1999.

[7] S. Sen, L. Gao, J. Rexford, and D. Towsley, "Optimal patching schemes for efficient multimedia streaming," in *Proceedings of NOSSDAV'99*, 1999.

[8] J. Y. B. Lee, "UVOD — a unified architecture for video-on-demand services," *IEEE Communication Letters*, vol. 3, pp. 277–279, Sept 1999.

[9] D. L. Eager, M. K. Vernon, and J. Zahorjan, "Optimal and efficient merging schedules for video-on-demand servers," in *Proceedings ACM Multimedia 99*, (Orlando, FL), pp. 199–202, 30 Oct.-5 Nov. 1999.

[10] S. Sheu, K. A. Hua, and W. Tavanapong, "Chaining: A generalized batching technique for video-on-demand systems," in *Proceedings of IEEE International Conference on Multimedia Computing and Systems*, (Ottawa, Canada), pp. 110–117, June 1997.

[11] K. C. Almeroth and M. H. Ammar, "The use of multicast delivery to provide a scalable and interactive video-on-demand service," *IEEE Journal on Selected Areas in Communications*, vol. 14, pp. 1110–1122, August 1996.

[12] A. Dan, P. Shahabuddin, D. Sitaram, and D. Towsley, "Channel allocation under batching and VCR control in video-on-demand systems," *Journal of Parallel and Distributed Computing*, vol. 30, no. 2, pp. 168–179, 1995.

[13] W.-F. Poon and K.-T. Lo, "Design of multicast delivery for providing VCR functionality in interactive video-on-demand systems," *IEEE Transactions on Broadcasting*, vol. 45, pp. 141–8, March 1999.

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

[15] 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, pp. 343–346, May 1998.

[16] C. C. Aggarwal, J. L. Wolf, and P. S. Yu, "Design and analysis of permutation-based pyramid broadcasting," *ACM/Springer Multimedia Systems*, vol. 7, no. 6, pp. 439–448, 1999.

[17] K. A. Hua and S. Sheu, "Skyscraper broadcasting: a new broadcasting scheme for metropolitan video-on-demand systems," *ACM Computer Communication Review*, vol. 27, pp. 89–100, Oct 1997.

[18] L. Gao, J. Kurose, and D. Towsley, "Efficient schemes for broadcasting popular videos," in *Proceedings of NOSSDAV'98*, (Cambridge, UK), July 1998.

[19] Y.-C. Tseng, C.-M. Hsieh, M.-H. Yang, W.-H. Liao, and J.-P. Sheu, "Data broadcasting and seamless channel transition for highly-demanded videos," in *Proceedings of IEEE Infocom*, (Tel-Aviv, Israel), March 26-30 2000.

[20] S.-H. Yeung and S.-H. G. Chan, "Stream-bundling approach to provide video services over broadband networks," in *Proceedings of the Multimedia Computing and Networking (MMCN)*, (San Jose, CA), January 22-24 2001.

[21] E. Chang and H. Garcia-Molina, "MEDIC: A memory & disk cache for multimedia clients," in *Proceedings of the 6th IEEE International Conference on Multimedia Computing and Systems*, (Florence, Italy), pp. 493–99, June 1999. uses hard disk to extend a client's buffer economically, Minimizing client buffer cost.

[22] S.-H. G. Chan and F. Tobagi, "Distributed servers architecture for networked video services," *IEEE/ACM Transactions on Networking*, to appear.

[23] D. L. Eager, M. C. Ferris, and M. K. Vernon, "Optimized regional caching for on-demand data delivery," in *Proceedings of the 1999 IS&T/SPIE Conference on Multimedia Computing and Networking*, (San Jose, CA), pp. 301–316, Jan 1999.