Broadcasting Video With the Knowledge of User Delay Preference

S.-H. Gary Chan, *Member, IEEE* and S.-H. Ivan Yeung

***Abstract*—In designing a video broadcasting system, the delay preference of a user is traditionally regarded as unknown. In fact, such preference can be known upon user's arrival by employing some techniques such as i) delay-dependent charging, where users are offered different levels of pay-per-view (PPV) depending on the maximum delay they are willing to tolerate; or ii) reservation, where a user specifies the exact play-time of a movie in advance, and he/she is charged according to the length of the reservation period. We explore, for the first time, the impact of such delay knowledge on request scheduling and system cost in terms of user loss and stream requirement. For delay-dependent charging, we propose "Delay-Aware Broadcasting" (DAB) and its variant based on reservation (DAB-r), where allocation of server streams is driven by the delay tolerance of a user. DAB-r offers differentiated grade of services according to user PPVs (and thereof classes). As compared with a system where user delay preference is not known, our schemes achieve substantially lower user loss rate, higher revenue, and better fairness. Regarding reservation system, we consider a scheme where clients can pre-buffer video data. Unicast streams are used to merge requests back to the on-going broadcast streams. We show that a reservation system achieves substantially lower stream requirement as compared to an on-demand system based on "patching."**

***Index Terms*—Delay-aware broadcasting, delay preference, reservation scheme, stream requirement, video broadcasting.**

I. Introduction

RECENT advances in computing and communication technologies have made the provisioning of video services over networks a reality [1]. In order to offer cost-effective video services accommodating many users, broadcasting techniques can be used, where users requesting a certain content are served with a single stream. A typical video broadcasting system generally consists of 3 components: a central video server, a broadcast-capable network (such as cable networks) and the clients (Fig. 1). The central video server stores the video files and schedules requests, and delivers the requested videos to the users via the network. The broadcast-capable network is used so that multiple users may share a stream (or channel). The clients pay for their service. If their requested movies are not displayed within their delay expectation or at their specified time, they leave the system (i.e., renege) and hence are lost, constituting a decrease in revenue and service quality. There may also be a unicast network between the server and clients for communication and to reduce user delay. We will focus on such video systems in this paper.

Video systems are traditionally studied under the assumption that the delay preference of users is not known *a priori* [2]–[12]. As a matter of fact, user's delay preference can be known upon its arrival by employing some techniques. It is therefore of interest and importance to explore schemes on making use of such knowledge, and its implications on user loss and stream requirement.

One technique to know user delay preference is to use *delay-dependent charging*, in which a movie has different levels of pay-per-view (PPV) corresponding to maximum delay a user would experience. By choosing a particular PPV, the user's delay tolerance is revealed to the server. In general, the longer a user is willing to wait, the lower is his PPV.[1] Given the delay preference of each user, we propose a batching scheme termed "Delay-Aware Broadcasting" (DAB) so that the server streams can be allocated more effectively. In this scheme, the video server serves the user when it is about to renege (i.e., at its maximum waiting tolerance), along with the other requests for the same movie in the queue. We consider that a user not served by his deadline is regarded as lost. (Note that a user can certainly keep waiting instead of leaving the system once the maximum delay corresponding to the PPV is exceeded; however, we treat such case as good as a "user loss.") We show that DAB indeed achieves substantially lower user loss rate as compared to schemes where delay preference is not known. However, it does not offer a proper differentiated grades of services (in terms of loss rate) to the users depending on their PPVs. To address this, we propose a reservation scheme termed DAB-r in which streams can be reserved or re-allocated to those higher-priority users. We show that the scheme achieves appropriate differentiation among user classes, substantially lower loss rate, higher revenue, and better fairness in loss rate across movies. One strength of DAB and DAB-r is that, as opposed to some recently proposed schemes, there is no additional requirement on client buffers or server caches to achieve a substantially lower loss rate. To put in other words, our schemes are independent with other client buffering or server caching techniques such as "patching" to further reduce the loss rate.

Manuscript received April 5, 2002; revised March 10, 2003. This work was supported in part by the Areas of Excellence (AoE) Scheme on Information Technology funded by the University Grant Council in Hong Kong (AoE/E-01/99), and by the Competitive Earmarked Research Grant of the Hong Kong Research Grant Council (HKUST6014/01E).

S.-H. G. Chan is with the Department of Computer Science, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong (e-mail: gchan@cs.ust.hk).

S.-H. I. Yeung is with the Department of Electrical and Electronic Engineering, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong (e-mail: gchan@cs.ust.hk).

Digital Object Identifier 10.1109/TBC.2003.813438

[1] By setting the PPV pricing structure, it is hence able to alter user's waiting behavior. It is beyond the scope of this paper to address how user delay behavior can be altered by means of pricing/charging schemes, which by itself is an active area in marketing research.

Another way to know user's delay preference is through *reservation*, in which users specify to the server the exact play-time of the movies in advance. The reservation period, defined as the difference between the play-time and the arrival time, is hence the waiting tolerance of the user. Given the reservation period, video can then be pre-fetched in advance by means of client buffering. We consider a scheme where movies are broadcast in a staggered fashion. If a user reserves a movie at a specific time beyond the beginning of an upcoming broadcast point, the client can prefetch the video into a buffer when the broadcast stream begins; in this way, the client can playback the movie strictly out of its buffer.[2] On the other hand, if the user reserves the movie at a time before the beginning of the next broadcast point, a unicast stream and an on-going broadcast stream are used to serve the user. Clearly, if users do not want to wait for their movies, such a reservation system reduces to an on-demand one. Note that the PPV pricing structure can be based on reservation period and hence can alter user delay preference. We address in this paper given a certain reservation behavior, how system parameters pertaining to buffer size and broadcast interval can be designed to minimize the stream requirement. We demonstrate that a reservation system can substantially reduce the number of streams required (by many times in our example) as compared to an on-demand system based on "patching."

We briefly discuss previous work as follows. One of the earliest batching schemes is called "First-Come-First-Serve" (FCFS) [2]. The scheme is based on no knowledge on user delay preference. All requests join a single queue. Whenever a stream is available, the request at the head of the queue, together with all the other requests for the same video, is served. Besides FCFS, other schemes such as "Maximum Queue Length" (MQL), "Max_Batch" and "Min_Idle" have also been proposed in [2], [8], [10]. All these schemes are based on *partial* user delay information (such as knowing the minimum waiting time of the users) without prioritization on user classes. We differ from them by considering schemes based on *full* knowledge of user delay behavior, and show that such knowledge achieves substantially lower loss rate or stream requirement. We also study how users may be prioritized to offer differentiated grades of services. A policy based on the knowledge of user delay tolerance termed LAMB (LookAhead Maximize-Batch) has been proposed in [13]. However, LAMB is quite complex and requires continuous computationally-intensive optimization in the server, and has no mechanism to differentiate user classes. Client buffering techniques as used in video broadcasting have also been discussed extensively in [14]–[23]. All these schemes consider meeting a certain fixed user delay requirement, instead of meeting a heterogeneous reservation schedule. A more recent scheme termed "patching" (also known as "catching" or "stream tapping") is proposed and studied in [24]–[31], which is an on-demand scheme making use of broadcasting and client buffering. As compared with it, our reservation system can achieve much lower bandwidth requirement.

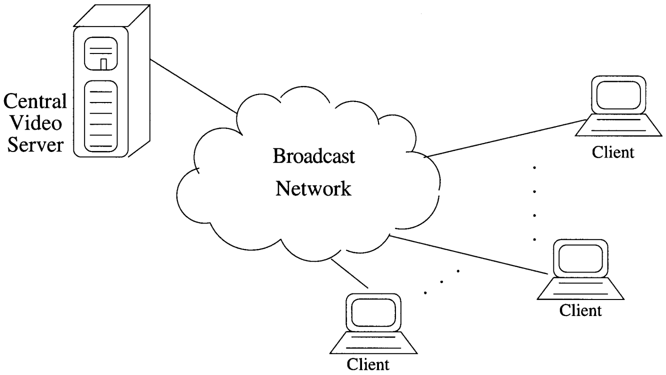


Fig. 1. A video broadcasting system.

This paper is organized as follows. We first describe delay-aware broadcasting and present some illustrative simulation results in Section II. The operation of the reservation system is described and analyzed in Section III. We conclude in Section IV.

II. The Delay-Aware Broadcasting Schemes

We first present delay-aware broadcasting (DAB) and its reservation variant, DAB-r, in Section II-A, followed by some illustrative simulation results to illustrate their performance in Section II-B.

A. Scheme Description

In Delay-Aware Broadcasting without channel reservation (DAB), per-movie queue is used, i.e., a user joins a queue corresponding to the movie it requests after revealing its maximum delay tolerance to the server. The server keeps a sorted list of the reneging time of the users, and serves the one just about to leave, along with all the requests for the same movie. If there is no available server channel at a reneging point, the user is not served and is lost. We show the operation of this scheme in Fig. 2. In this figure, user 1 and user 3 request a movie (say, movie A), while user 2 and user 4 request another movie (movie B). When user 1 is about to leave (i.e., renege 1), both user 1 and user 3 are served as a batch if there is available channels; otherwise, user 1 is lost.

In general users who pay more (those premium users) are those with low delay preference and expect lower loss rate (i.e., lower probability of missing their delay deadlines). DAB as described above does not differentiate users according to their delay preference or PPVs. In order to appropriately differentiate user classes in terms of their delay preference and assign server channels accordingly, we can modify DAB with channel reservation (DAB-r). In this scheme, all video requests join a single queue and there are N different priority level for the users (depending on their PPV). If there is a channel available, the server reserves the channel to the users in a FCFS manner according to a probability depending on the user priority class. This reserved channel ensures that the user, and hence the movie batch, can be served at its reneging time. Clearly, by adjusting the reservation probability, the chance of channel run-out for that priority group of users can be changed accordingly, hence achieving differentiated loss rate and services.

[2] We distinguish "user" from "client" here, where a "user" makes movie request and is the money-payer in the system, while a "client" is a desktop computer or TV set-top box which prefetches data on user's behalf.

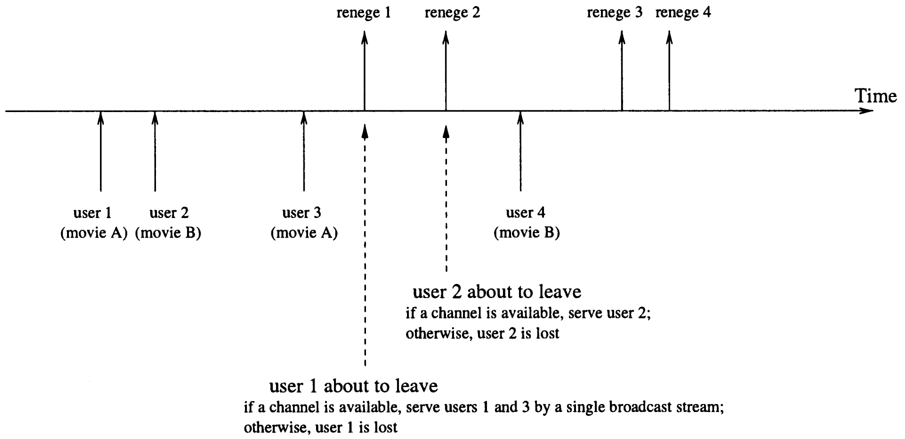

renege 1
renege 2
renege 3
renege 4
Time
user 1 (movie A)
user 2 (movie B)
user 3 (movie A)
user 4 (movie B)
user 2 about to leave
if a channel is available, serve user 2;
otherwise, user 2 is lost
user 1 about to leave
if a channel is available, serve users 1 and 3 by a single broadcast stream;
otherwise, user 1 is lost

Fig. 2. The operation of delay-aware broadcasting (DAB).

Note that a premium user may arrive to the system when all channels are already reserved to movies with lower-priority users. In order to ensure that higher-priority users enjoy better chance in channel allocation, "channel stealing" or channel reallocation is implemented as follows. When a user arrives and there is no available channel, the system checks whether there are channels already reserved for those movies with lower-priority users only. If so, the channel corresponding to the one with the fewest requests is re-allocated (and hence "stolen") to the newly arrived user with a certain re-allocation probability p ($p = 1$ means that the channel is always re-allocated). If all channels are reserved for the higher-priority users, no channel can be re-allocated. Clearly, there is a trade-off between p and loss rate: the higher the p is, the lower is the loss rate of the users with high priority. We choose the movie with the fewest requests because of its simplicity and to minimize the number of users affected lest the channel is re-allocated.

In studying the schemes, we are interested in the following metrics:

- Overall loss probability P_L, and the loss probability for class k users $P_{L,k}$: P_L is defined as the ratio of the total number of user lost to the total number of arrivals in the system, and $P_{L,k}$ is defined as the ratio of the total number of class k users lost to the total number of class k arrivals in the system. The lower k is, the higher is the priority of the user;
- Average revenue rate R_v ($/min), defined as the total PPV collected from the served users divided by the total time examined;
- Unfairness, which is defined by the variance of loss rate of the movies divided by the total number of movies in the system [2], i.e.,

$$\text{Unfairness} = \frac{\sum_{m=1}^{M} (P_L(m) - P_L)^2}{M}, \tag{1}$$

  where $P_L(m)$ is the loss rate for movie m (given by the ratio of the total number of lost requests for movie m with respect to the total number of requests for that movie). Clearly, unfairness increases when there is discrimination of requests among movies; and
- Overall average delay D (in minutes), and average delay in each class D_k minutes, defined as the average delay from the time user enters the system until the time the movie is displayed.

B. Illustrative Simulation Results

We present in this section the simulation results for the performance of the DAB and DAB-r schemes, and compare them with FCFS. In our study, we consider that, as in other literature, user arrives to the system according to a Poisson process with rate λ req/min. There are $M = 100$ movies of length $T_h = 120$ minutes in the system. The popularity of the movies is according to a Zipf distribution, with $\lambda_i \propto 1/i^{\zeta}$, where $\zeta = 0.746$ [2]. The server has $N_s = 400$ video streams. We consider a system with 3 classes and delay-dependent PPV. Class 1 users are the highest priority users, whose maximum delay tolerance is 5 minutes and service charge is 4 units (or dollars). Class 2 users have maximum delay tolerance of 10 minutes with 2 units of service charge, and Class 3 users have 20 minutes delay tolerance with 1 unit of service charge. Class 1 users should be offered the highest service quality (i.e., the lowest loss rate) as compared with the other user classes. The fraction of users in class 1, 2 and 3 are 10%, 30% and 60%, respectively. We consider that channels, if available, are always reserved to Class 1 and Class 2 users in the DAB-r scheme, and a re-allocation probability p.

We first consider user loss rate for DAB-r. We show in Fig. 3 the overall loss rate P_L and the loss rate for each class $P_{L,1}$, $P_{L,2}$, $P_{L,3}$ with respect to the channel re-allocation probability p, for an arrival rate $\lambda_0 = 10$ req/min. As expected, the loss rate for class 1 decreases while that of class 2 increases as p increases, since reserved channels of this class are more likely to be re-allocated to class 1 users. $P_{L,3}$ does not change much since p affects only class 1 and class 2 users. If we want to maintain the proper relationship $P_{L,1} < P_{L,2} < P_{L,3}$, we can choose $p > 0.15$ in this case. We would choose $p = 0.8$ in our following simulation since it achieves a low $P_{L,1}$ (about 1%) and a relatively low $P_{L,2}$ (about 6%). Note that the overall loss rate is given by $P_L = \alpha_1 P_{L,1} + \alpha_2 P_{L,2} + \alpha_3 P_{L,3}$, where α_1, α_2, and α_3 are the fractions of class 1, class 2, and class 3

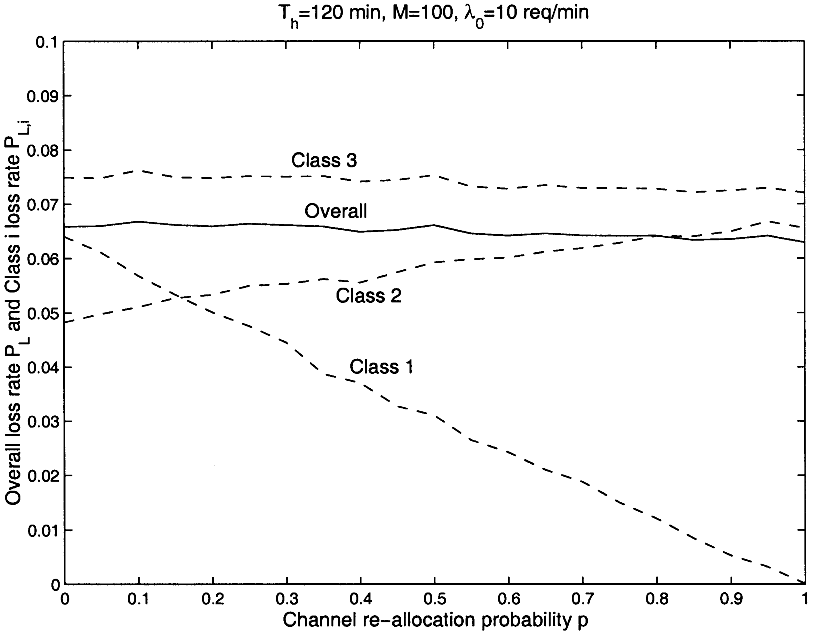


Fig. 3. Overall loss probability P_L and class loss probability $P_{L,\,i}$ versus channel re-allocation probability p for DAB-r.

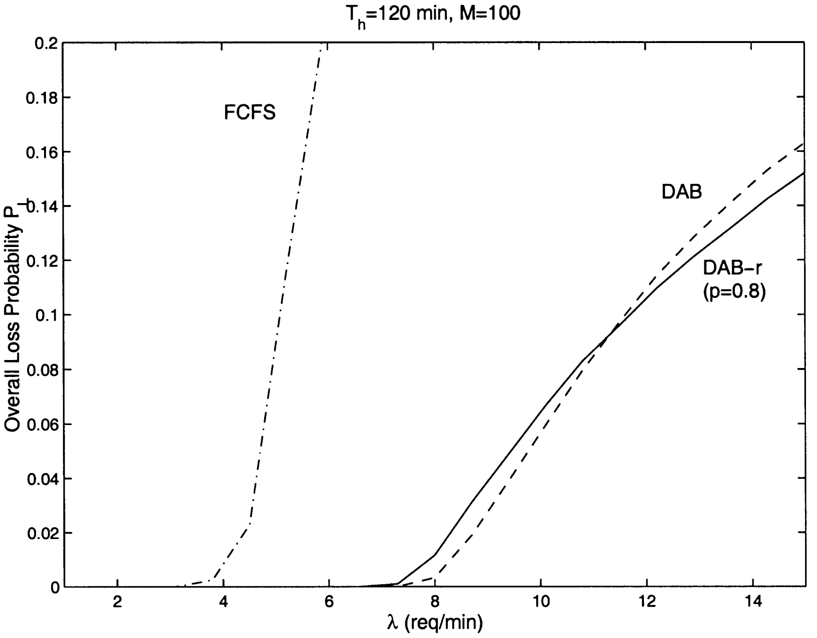


Fig. 4. Overall loss probability P_L versus λ for FCFS, DAB, and DAB-r.

users, respectively (=0.1, 0.3, and 0.6 here, respectively). The overall loss rate P_L does not depend much on p, and decreases only slightly as p increases.

We compare in Fig. 4 P_L with respect to λ for FCFS, DAB and DAB-r ($p = 0.8$). As λ increases, the loss probability increases, as more requests compete for the limited number of streams. Clearly, both DAB and DAB-r achieve much lower loss rate than FCFS, showing that making use of the knowledge of user delay preference can reduce the loss rate significantly. Note that though DAB and DAB-r perform similarly in terms of overall loss rate, this is not so when we examine the loss rate in each class, as shown in the following figures.

We compare in Figs. 5 and 6 $P_{L,\,1}$ and $P_{L,\,3}$ with respect to λ for FCFS, DAB and DAB-r. The loss rate of FCFS is always much higher than both DAB and DAB-r, because it does not consider user delay tolerance and allocates channels to users too soon. It is worth noting that in DAB, the class 1 users (i.e., those premium users who pay more for a lower delay) has an even higher loss rate than those class 3 users. DAB-r, on the other hand, achieves a much more proper differentiation of services across user classes: The class 1 requests are served with a higher priority, thus their loss probability is lower than that of class 3 users. $P_{L,\,3}$ for DAB-r is higher than that of DAB, due to the fact that channels are reserved for class 1 and class 2 users, leading

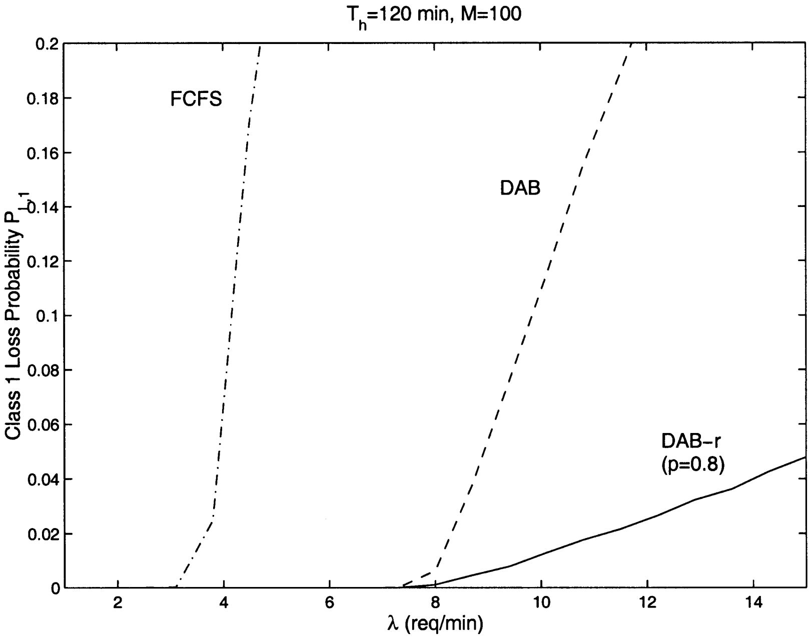


Fig. 5. Class 1 loss probability $P_{L,1}$ versus λ for FCFS, DAB, and DAB-r.

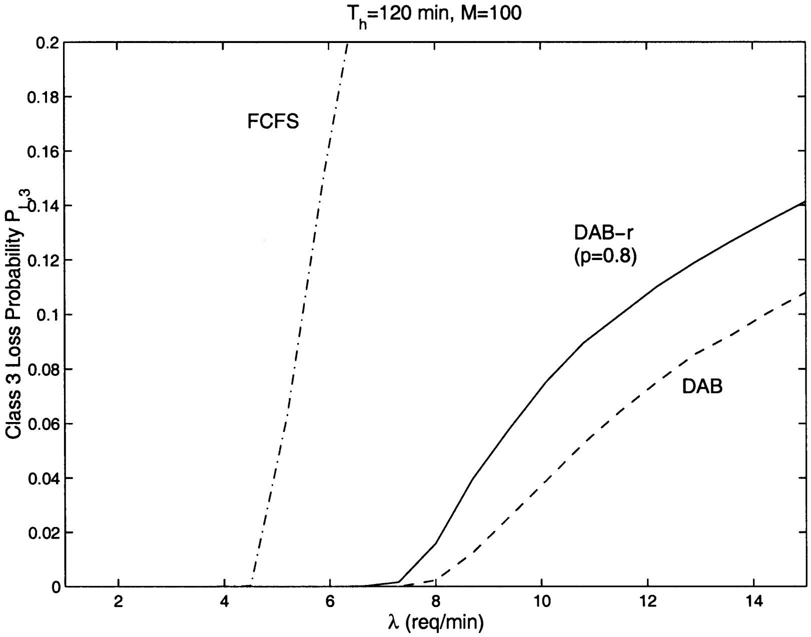


Fig. 6. Class 3 loss probability $P_{L,3}$ versus λ for FCFS, DAB, and DAB-r.

to fewer channels for serving class 3 users. Note that $P_{L,3}$ of both DAB and DAB-r does not increase rapidly with λ, due to batching effect and the fact that the users can tolerate longer delay than the other two classes.

We next plot in Fig. 7 the overall revenue R_v for DAB, DAB-r and FCFS with respect to λ. When λ increases, the revenue first increases rather linearly (since there is little user loss) and then gradually flattens off (due to user loss). Clearly, R_v for both DAB and DAB-r is significantly higher than that of FCFS, due to the difference between their loss rates. DAB-r has a slightly higher R_v because more class 1 users are served. The performance of DAB and DAB-r is very close to an ideal system (which has no user loss) characterized by $\lambda\times$ (average PPV).

We show in Fig. 8 the unfairness in loss rate vs. λ between the schemes FCFS and DAB-r. We clearly see that DAB-r achieves significantly better fairness. The unfairness for FCFS increases steeply as λ increases beyond a point (corresponding to the point at which the loss rate increases quickly). This is because as λ increases, the server begins to run out of channels and queue starts to built up. Since the request rate for more popular movies is higher than those unpopular ones and a released server channel is assigned according to the FCFS policy, the requests for popular movies are more likely to be assigned a channel, leading

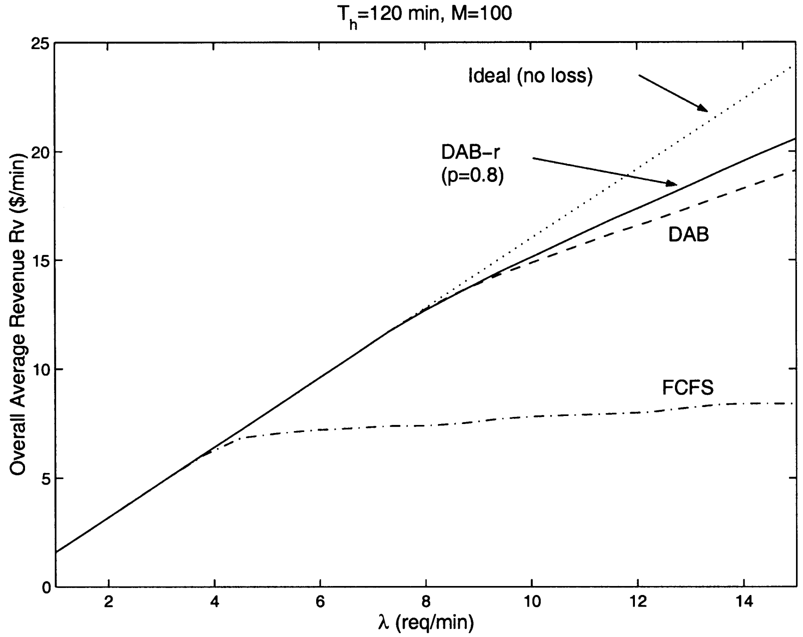


Fig. 7. Overall revenue R_v versus λ for FCFS, DAB, and DAB-r.

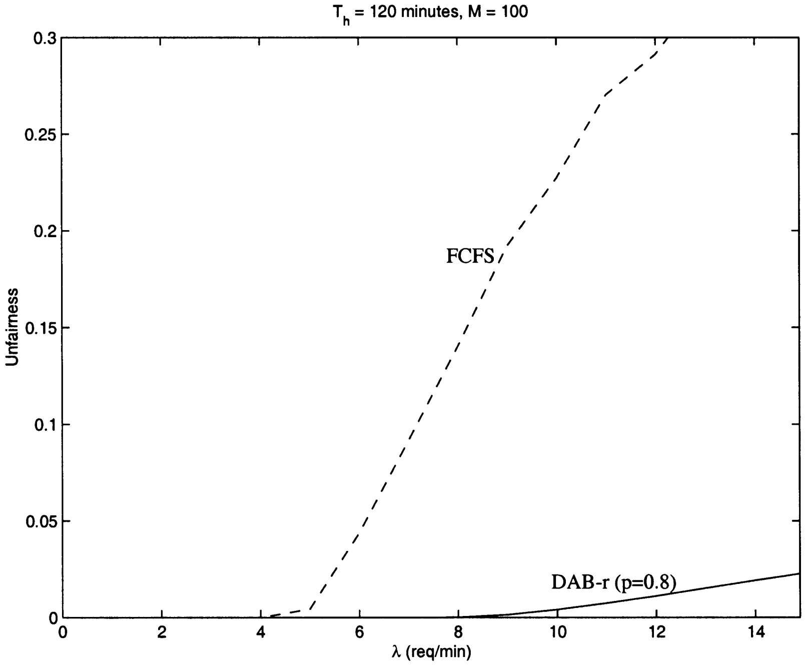


Fig. 8. Unfairness versus λ for FCFS and DAB-r.

to unfairness (nonuniformity in loss rate across movies). On the other hand, since DAB-r (and as well as DAB) is based on user's reneging time, and not on movie popularity nor user request rate, its fairness is much better than that of FCFS scheme.

We finally show the overall average delay D with respect to λ in Fig. 9 for FCFS, DAB and DAB-r. Note that D of both DAB and DAB-r schemes is higher than that of FCFS. This is expected because DAB and DAB-r make better use of the streams by allocating the streams to users at their points of reneging; therefore channels are not allocated too fast and hence run out too soon. FCFS, on the other hand, allocates streams to users as soon as they arrive, causing the streams to run out too fast. The higher delay of DAB and DAB-r should not be a concern because the delay expectation of the users are mostly met: they are willing to wait for such a time before the movies start. For DAB and DAB-r, D decreases with increasing λ due to batching effect: the later arrivals in a batch enjoy lower delay, which brings down the average delay. In FCFS, D is low for small λ (due to little queuing time) but increases quite fast as λ increases because channels quickly run out. The increase is tamed as λ further increases due to batching effect as mentioned above. Regarding the average delay of the three individual classes, we find

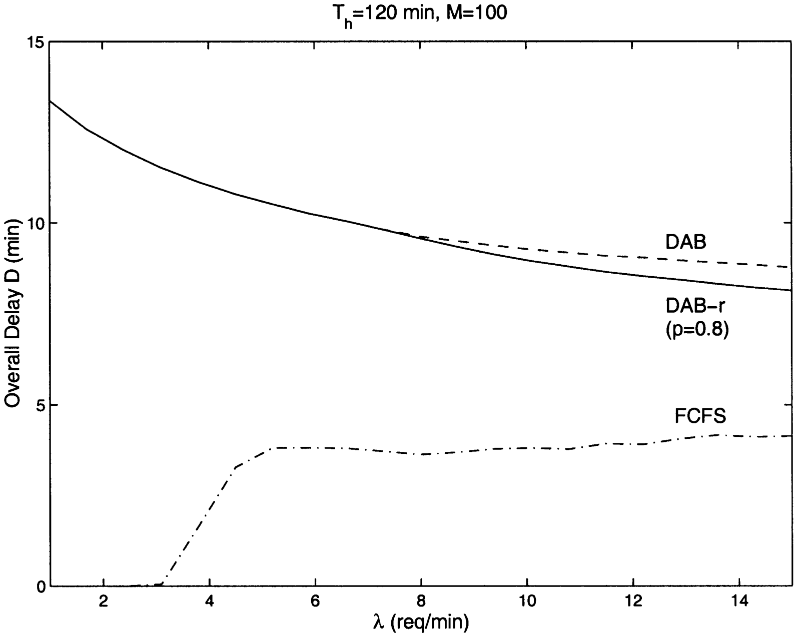


Fig. 9. Overall average start-up delay D_{av} versus λ for FCFS, DAB, and DAB-r.

that they exhibit the same trend, with the average delay for each class being lower than each of their respective maximum delay tolerance.

III. A Reservation System

In this section, we present a reservation scheme to achieve lower system requirement by means of client buffering. We first describe the scheme in Section III-A, and then show its analysis and optimization in Section III-B. We finally present some illustrative numerical results in Section III-C.

A. Scheme Description

In the reservation system, the server channels are divided into broadcast streams and unicast streams. A movie is broadcast in a staggered fashion to its completion using a broadcast stream every W minutes. Therefore, the number of broadcast streams allocated for the movie is $\lceil T_h/W \rceil$, where T_h is the movie length (holding time) in minutes. Requests with the display time beyond the upcoming broadcast point can be served solely by the broadcast stream with prefetching. However, for a request with the display time falling before the upcoming broadcast time, a unicast stream and prebuffering are used to merge the request back to the (temporally) closest broadcast stream.

We show in Fig. 10 the operation of this scheme, where users specify the exact starting time of the movie when they arrive. Let D be the reservation period of the user, and x be the time between the arrival and the next broadcast point. We need to consider two cases:

a) $x \leq D$ (i.e., the movie is scheduled to be displayed after the start of the next broadcast stream): the client pre-buffers the video from the broadcast stream right before D. At the display time, the client simply plays back the movie from the buffer.

b) $x > D$: At the time of arrival, the client first pre-buffers the video data from the ongoing broadcast stream which is started right before the arrival. When the movie starts playing, a unicast stream is allocated to supply the beginning portion of the video for a (short) period of time equal to $W - x$ minutes, after which the unicast stream is relinquished and the video data is streamed from its own buffer.

We clearly see that the client buffer is no more than W minutes of video time. The value of W can be optimized in order to minimize the total number of unicast and broadcast streams, given a certain user reservation behavior. Note that the special case $D = 0$ corresponds to an on-demand system and the technique is called "patching." We compare the bandwidth requirement of patching with this reservation system.

B. Scheme Analysis

We present the analysis on the reservation system in this section. If the bandwidth is sized appropriately, the operation of a movie is independent of the others, and hence it is sufficient to focus on a particular movie with its request being Poisson with rate λ req/min. We are interested in the following parameters:

- The number of streams required for a movie, S: This is the sum of the broadcast streams and unicast streams used. Clearly, if W is long, the number of unicast streams used is large and hence S is large; on the other hand, if W is short, the number of broadcast streams used is large and hence S is large. Therefore, there is an optimal broadcast interval W^* to minimize S. We would like to obtain such W^*, and compare S^* (the optimal S) with that of an optimized on-demand system based on patching; and
- The buffer requirement, B: As mentioned before, the buffer requirement is $B = W$.

In the reservation system, the number of broadcast streams required is clearly T_h/W (ignoring the nonintegral part). Re-

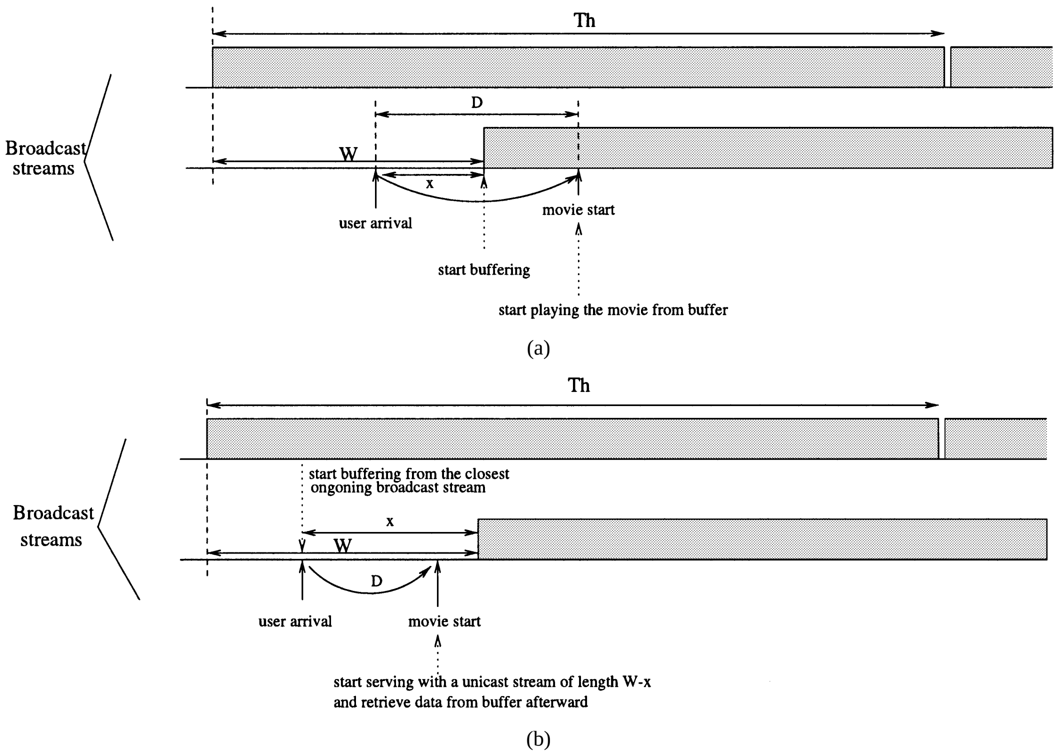

Th
D
Broadcast streams
W
x
user arrival
movie start
start buffering
start playing the movie from buffer
(a)
Th
start buffering from the closest ongoning broadcast stream
Broadcast streams
x
W
D
user arrival
movie start
start serving with a unicast stream of length W-x and retrieve data from buffer afterward
(b)

Fig. 10. Operation of the reservation system. (a) Movie starts after the next upcoming broadcast point. (b) Movie starts before the next upcoming broadcast point.

garding the number of unicast streams, we only need to consider the case $D < x$, since no unicast stream is used otherwise. Assume that D follows a certain distribution given by $P(D \leq d) = R(d)$, independent of other users. The probability that a random user arriving x minutes before the next broadcast point chooses a time before that point is obviously given by $P(D \leq x) = R(x)$, and the corresponding length of the unicast stream required is $(W - x)$. Moreover, given an arrival within the interval W, the time at which the user arrives is uniformly distributed and hence the probability that the user arrives at time $[x, x + dx]$ is dx/W (Poisson property). Therefore, the average length of the unicast streams is $\int_0^W (W - x)R(x)(dx/W)$, and the average number of unicast streams required, U, is given by (by Little's formula)

$$U = \lambda \int_0^W (W - x)R(x)\frac{dx}{W}, \tag{2}$$

which yields

$$S = \frac{T_h}{W} + \lambda \int_0^W (W - x)R(x)\frac{dx}{W}. \tag{3}$$

To minimize S, we differentiate it with respect to W and set it to zero to obtain W^*.

Note that in the traditional on-demand system based on patching, users cannot tolerate any delay and hence $R(x) = u(x)$, where $u(x)$ is a unit-step step function given by $u(x) = 0$ when $x < 0$ and $u(x) = 1$ when $x \geq 0$. Thus, the average number of unicast streams required is given by $U = \lambda \int_0^W (W - x)\,dx/W = \lambda W/2$, which yields $S = T_h/W + \lambda W/2$. We hence have

$$W^* = \sqrt{\frac{2T_h}{\lambda}}, \tag{4}$$

and

$$S^* = \sqrt{2\lambda T_h}. \tag{5}$$

C. Illustrative Numerical Examples

We present in this section the performance of the reservation system and compare it with the on-demand patching system. As an example, we consider that the reservation period D is exponential with mean $1/\mu$ minutes [i.e., $R(d) = 1 - e^{-\mu d}$], yielding

$$S = \frac{T_h}{W} + \lambda W\left(\frac{1}{2} - \frac{1}{\mu W} + \frac{1}{\mu^2 W^2}\left(1 - e^{-\mu W}\right)\right), \tag{6}$$

from which S^* and W^* can be obtained. We consider a baseline system where $T_h = 100$ minutes, $\lambda_0 = 50$ req/min and $\mu = 1/30$ min^{-1}.

We plot in Fig. 11 S with respect to W. As W increases, S decreases quickly at the beginning (due to a decrease in broadcast streams required) and then rises again (due to an increase in unicast streams). S clearly has a minimum at a certain W^*. We show in Fig. 12 W^* with respect to λ for the reservation and on-demand system (which also corresponds to client buffer requirement). As λ increases, W^* decreases, so as to take advantage of the batching effect by means of broadcasting and to decrease the requirement of unicast streams. We see that the buffer requirement as given by W^* is not high in most of the cases ($\leq$10% of the movie length). The reservation system has a larger W^* as compared with the on-demand system, mainly due to the fact that unicast streams are less likely to be used in the reservation system.

We show in Fig. 13 S^* and the corresponding requirement of broadcast channels with respect to λ, for the reservation and on-demand systems. The number of streams used in

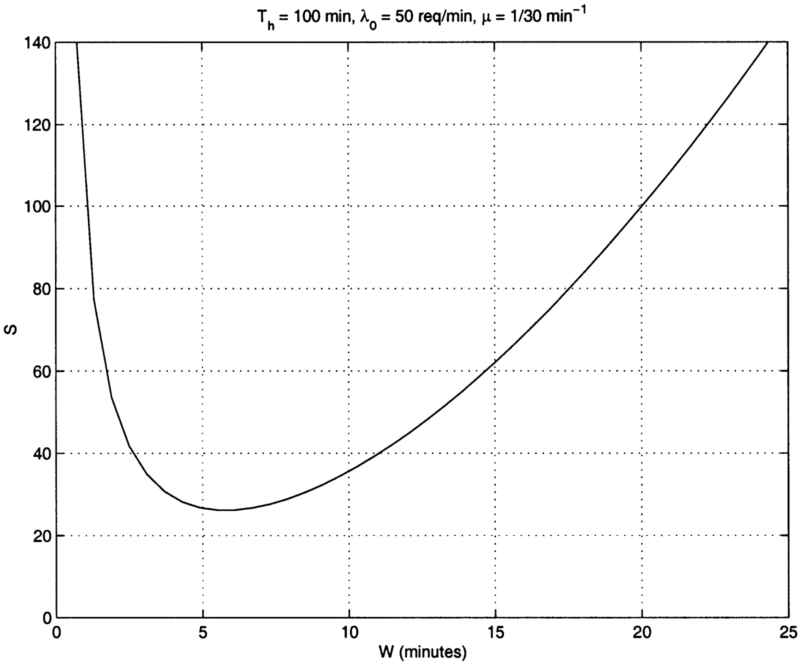


Fig. 11. S versus W for the reservation system.

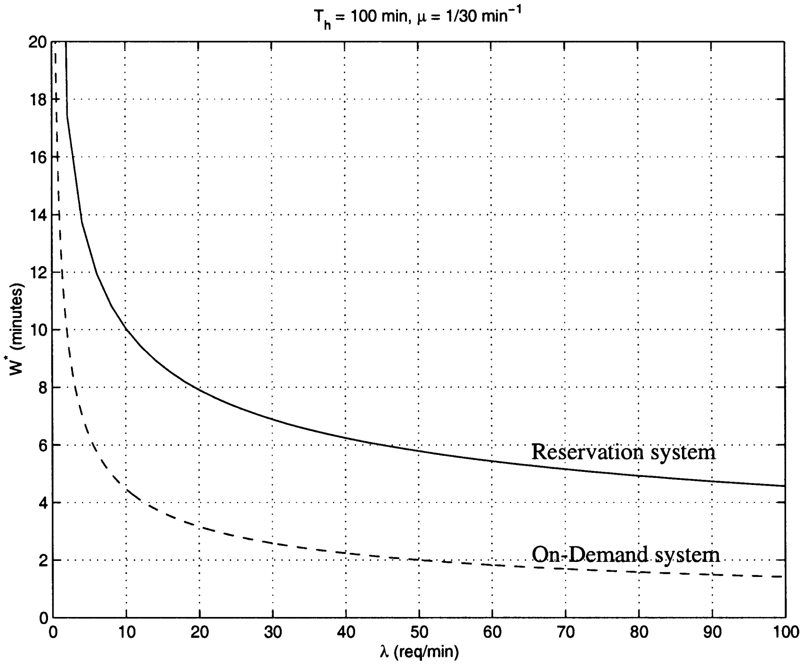


Fig. 12. W^* versus λ for the reservation system and an on-demand system based on patching.

the reservation system is substantially lower than that of the on-demand system (by 3–4 times), especially when the request rate is high. The requirement of broadcast channels for the reservation system is also substantially lower, due to its larger W^*. From the difference between S^* and the number of broadcast channels, we see that the number of unicast streams used in reservation system is also significantly lower than that of the on-demand system.

We finally show the effect of user reservation period (i.e., user waiting tolerance) $1/\mu$ on S^* and W^*, given λ_0, in Fig. 14. Clearly, W^* decreases when the average reservation period $1/\mu$ increases. This is because users are more willing to wait and hence the broadcast interval can be longer. As $1/\mu$ increases, S^* decreases, mainly because more and more users are served by broadcast streams rather than unicast streams. Note that the number of unicast streams required is quite low even with such a high arrival rate (when $1/\mu \geq 20$ minutes, fewer than ten unicast streams are required as in Fig. 13). Note that the case $1/\mu = 0$ corresponds to the on-demand system, which has a much higher bandwidth requirement.

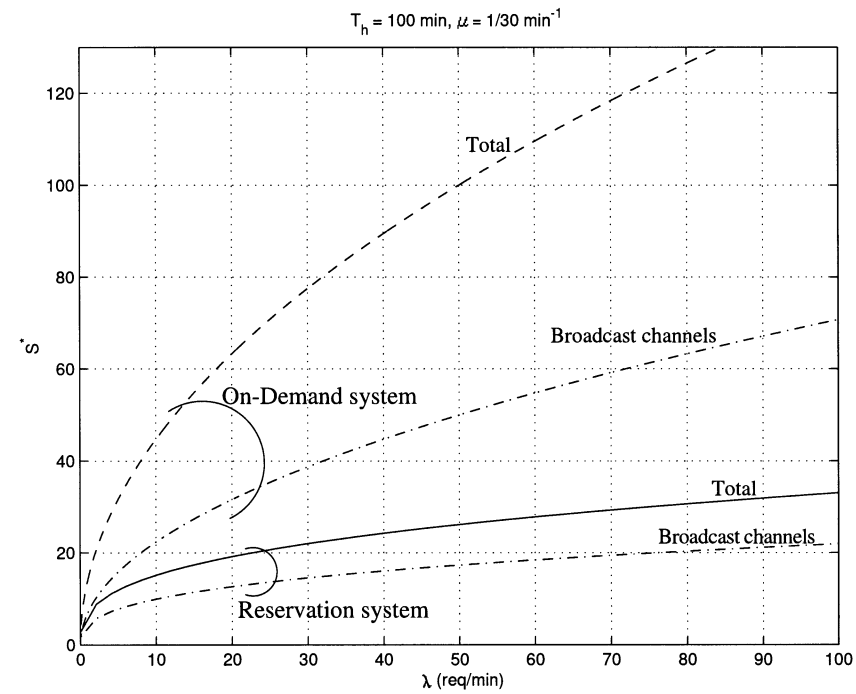


Fig. 13. S^* versus λ for the reservation system and on-demand system.

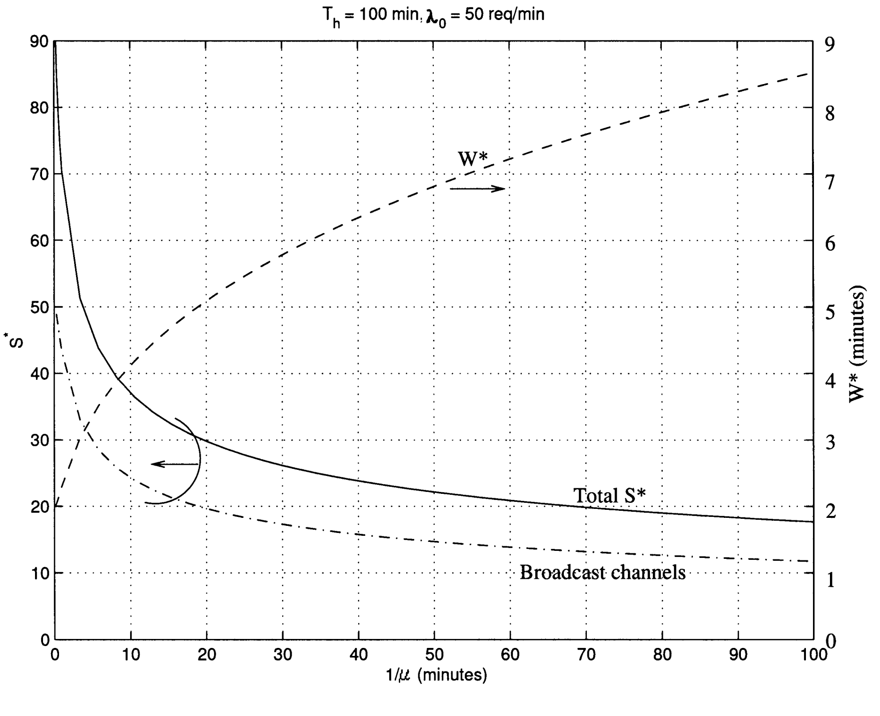


Fig. 14. S^* and W^* versus $1/\mu$ for the reservation system.

IV. Conclusions

Traditional study on video systems assumes that user delay preference is unknown. In this paper, we consider that such delay preference is revealed to the server upon users' arrival. We have study the case for a system based on delay-dependent charges and a reservation system.

Regarding delay-dependent charges, the PPV of a movie depends on the maximum delay users are willing to tolerate. We have studied "Delay-Aware Broadcasting" (DAB), which serves users at their reneging point, along with all the other waiting users for the same movie. Via simulation, this scheme is shown to achieve substantially lower user loss rate (and hence higher revenue) than a FCFS policy in which user delay preference is not taken into consideration. However, DAB is found to have a high loss rate for those low-delay users. To address this and to offer differential grade of services to users, we have studied a reservation variant termed "DAB-r" in which freed channels can be reserved for the higher-priority arrivals. In the scheme, some of the reserved channels can also be re-allocated to those higher-priority users when bandwidth is limited. With DAB-r, we show that a proper differentiation of services can be offered to the users.

Regarding reservation systems, we have analyzed a system in which users specify in advance the exact time their movies are to be displayed and are charged according to their reservation period. In the scheme, a movie is broadcast in a staggered fashion. If the reserved play-time of a movie is beyond the next upcoming broadcast point, the client simply buffers the video at the broadcast point and playbacks from its buffer at the due time. Otherwise, the client immediately pre-buffers the (temporally) closest ongoing broadcast stream, while a unicast stream is used to supply the missing startup portion of the movie until the client is able to retrieve the matched video data from its buffer. We have considered how the total number of streams can be minimized by optimizing the broadcasting interval, and show that such a scheme, as compared with a traditional on-demand system based on patching, can substantially reduce the server bandwidth (by 3 to 4 times in our examples) with very little client buffering ($\leq$20 minutes). Our study shows that making use of the knowledge of user delay preference can lead to a system with low cost and low loss. We are currently looking into some fundamental issues in video broadcasting. For example, given a limited number of channels and user delay tolerance in the system, what is the best way to allocate channels so as to minimize user loss? The answer to this question will serve as an important bound for various batching algorithms and shed lights on how requests should be scheduled when user delay tolerance is not completely revealed to the system.

REFERENCES

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

[2] A. Dan, D. Sitaram, and P. Shahabuddin, "Dynamic batching policies for an on-demand video server," *ACM/Springer Multimedia Systems*, vol. 4, no. 3, pp. 112–121, June 1996.

[3] A. D. Gelman and S. Halfin, "Analysis of resource sharing in information providing services," in *Proc. IEEE Globecom'90*, 1990, pp. 312–316.

[4] V. O. K. Li and W. Liao, "Distributed multimedia systems," *Proc. IEEE*, vol. 85, no. 7, pp. 1063–1108, July 1997.

[5] V. O. K. Li, W. Liao, X. Qiu, and E. W. M. Wong, "Performance model of interactive video-on-demand systems," *IEEE J. Select. Areas Commun.*, vol. 14, no. 6, pp. 1099–1109, Aug. 1996.

[6] S.-H. G. Chan and F. Tobagi, "Trade-off between system profit and user delay/loss in providing video services with request batching," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 11, no. 8, pp. 916–927, Aug. 2001.

[7] 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, June 1996, pp. 253–258.

[8] C. C. Aggarwal, J. L. Wolf, and P. S. Yu, "The maximum factor queue length batching scheme for video-on-demand systems," *IEEE Trans. Comp.*, vol. 50, no. 2, pp. 97–110, Feb. 2001.

[9] A. K. Tsiolis and M. K. Vernon, "Group-guaranteed channel capacity in multimedia storage servers," *Performance Evaluation Review*, vol. 25, no. 1, pp. 285–297, 1997.

[10] H. Shachnai and P. S. Yu, "Exploring wait tolerance in effective batching for video-on-demand scheduling," *ACM/Springer Multimedia Systems*, no. 6, pp. 382–394, 1998.

[11] K. C. Almeroth, "Adaptive workload-dependent scheduling for large-scale content delivery," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 11, no. 3, pp. 426–439, Mar. 2001.

[12] W.-S. Wen, S.-H. G. Chan, and B. Mukherjee, "Token-Tray/Weighted Queuing-Time (TT/WQT): An adaptive batching policy for near video-on-demand system," *Elsevier Computer Communications*, vol. 25, no. 9, pp. 890–904, June 1, 2002.

[13] N. L. S. da Fonseca and R. A. Façanha, "The look-ahead-maximize-batch batching policy," in *Proc. IEEE Globecom*, Rio de Janeiro, Brazil, Dec. 5–9, 1999, pp. 354–357.

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

[15] C. C. Aggarwal, J. L. Wolf, and P. S. Yu, "A permutation-based pyramid broadcasting scheme for video-on-demand systems," in *Proc. IEEE Int. Conf. Multimedia Computing and Systems*, Los Alamitos, CA, 1996, pp. 118–26.

[16] L.-S. Juhn and L.-M. Tseng, "Harmonic broadcasting for video-on-demand service," *IEEE Trans. Broadcast.*, vol. 43, no. 3, pp. 268–271, Sept. 1997.

[17] ——, "Fast data broadcasting and receiving scheme for popular video service," *IEEE Trans. Broadcast.*, vol. 44, no. 1, pp. 100–105, Mar. 1998.

[18] K. A. Hua and S. Sheu, "Skyscraper broadcasting: A new broadcasting scheme for metropolitan video-on-demand systems," *ACM Computer Communication Review*, vol. 27, no. 4, pp. 89–100, Oct. 1997.

[19] L. Gao, J. Kurose, and D. Towsley, "Efficient schemes for broadcasting popular videos," in *Proc. NOSSDAV'98*, Cambridge, UK, July 1998.

[20] J.-F. Pâris, S. W. Carter, and D. D. E. Long, "A hybrid broadcasting protocol for video on demand," in *Proc. 1999 IS&T/SPIE Conf. Multimedia Computing and Networking*, San Jose, CA, Jan. 1999, pp. 317–326.

[21] ——, "A universal distribution protocol for video-on-demand," in *Proc. Int. Conf. Multimedia and Expo*, New York, NY, July 30–Aug. 2, 2000, pp. 49–52.

[22] J.-F. Pâris, "An interactive broadcasting protocol for video-on-demand," in *Proc. 2001 IEEE Int. Performance, Computing, and Communications Conf.*, Phoenix, AZ, Apr. 4–6, 2001, pp. 347–53.

[23] S.-H. G. Chan and S.-H. I. Yeung, "Client buffering techniques for scalable video broadcasting over broadband networks with low user delay," *IEEE Trans. Broadcast.*, vol. 48, no. 1, pp. 19–26, Mar. 2002.

[24] J. Y. B. Lee, "UVoD—A unified architecture for video-on-demand services," *IEEE Commun. Lett.*, vol. 3, no. 9, pp. 277–279, Sept. 1999.

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

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

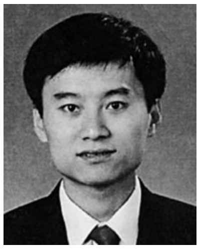
[27] L. Gao, Z.-L. Zhang, and D. Towsley, "Catching and selective catching: Efficient latency reduction techniques for delivering continuous multimedia streams," in *Proc. ACM Multimedia'99*, Orlando, FL, Oct. 30–Nov. 5, 1999, pp. 203–6.

[28] S. W. Carter and D. Long, "Improving video-on-demand server efficiency through stream tapping," in *Proc. Sixth Int. Conf. Computer Communications and Networks*, Las Vegas, NV, Sept. 22–25, 1997, pp. 200–7.

[29] S. W. Carter, D. Long, and J.-F. Pâris, "An efficient implementation of interactive video-on-demand," in *Proc. 8th Int. Symp. Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, San Francisco, CA, Aug. 29–Sept. 1, 2000, pp. 172–9.

[30] S. Ramesh, I. Rhee, and K. Guo, "Multicast with cache (mcache): An adaptive zero-delay video-on-demand service," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 11, no. 3, pp. 440–456, Mar. 2001.

[31] S.-H. G. Chan and F. Tobagi, "Distributed servers architecture for networked video services," *IEEE/ACM Trans. Networking*, vol. 9, no. 2, pp. 125–136, Apr. 2001.



S.-H. Gary Chan (S'89–M'98) received the Ph.D. degree in Electrical Engineering with a minor in Business Administration from Stanford University, Stanford, CA, in 1999, and the B.S.E. degree (highest honor) in Electrical Engineering from Princeton University, Princeton, NJ, in 1993.

He is currently an Assistant Professor with the Department of Computer Science, The Hong Kong University of Science and Technology, Hong Kong, and an Adjunct Researcher with the Microsoft Research Asia in Beijing. He was a Visiting Assistant Professor in networking at the Department of Computer Science, University of California, Davis, from September 1998 to June 1999. During 1992–1993, he was a Research Intern at the NEC Research Institute, Princeton, NJ. His research interest includes multimedia networking, peer-to-peer networks, high-speed and wireless communications networks, and Internet technologies and protocols.

Dr. Chan was a William and Leila Fellow at Stanford University during 1993–1994. At Princeton, he was the recipient of the Charles Ira Young Memorial Tablet and Medal, and the POEM Newport Award of Excellence in 1993. He is a member of Tau Beta Pi, Sigma Xi, and Phi Beta Kappa.

S.-H. Ivan Yeung received the Master of Philosophy from the Department of Electrical and Electronic Engineering in the Hong Kong University of Science and Technology (HKUST), Hong Kong, in 2000. He received the B.Eng. from the same University in 1998. From 2000 to 2001, he worked for SinoCDN Limited, Hong Kong, where he was a research staff and involved in conducting research on multimedia networks and content networking. He then worked as a Research Assistant at the Department of Electrical and Electronic Engineering in HKUST, where he conducted research on content networking and access network technologies. Currently, he is a Senior Engineer in SinoCDN Limited. His research interest includes multimedia networks, content delivery networks, content networking and Internet technologies.