# A Monotonic-Decreasing Rate Scheduler for Variable-Bit-Rate Video Streaming

Hin-lun Lai, Jack Yiu-bun Lee, *Senior Member, IEEE*, and Lian-kuan Chen, *Senior Member, IEEE*

***Abstract*—Variable-bit-rate (VBR) encoded videos can provide a more consistent visual quality than constant-bit-rate (CBR) encoded videos. However, the long-range bit-rate variations in VBR videos make it difficult to efficiently provide quality-of-service control in a video-on-demand system. Existing scheduling algorithms such as Optimal Smoothing, which requires both downward and upward bandwidth reallocations to adapt to the video bit-rate variations, simply cannot guarantee video delivery in networks with mixed video and data traffic. This study tackles this limitation by investigating a new scheduling algorithm with monotonic-decreasing rate allocations for scheduling video data transmissions. By eliminating upward bandwidth reallocations, the proposed scheduler can guarantee video delivery even in the presence of other data traffic in the same network. Moreover, results show that the proposed scheduler can achieve such performance guarantee without tradeoff in performance or resource requirements. This paper presents this new monotonic-decreasing rate scheduler, analyzes its fundamental properties, and evaluates its performance using a large number of real-world VBR video traces (274 DVD movies) in extensive trace-driven simulations.**

***Index Terms*—Performance guarantee, scheduling, smoothing, variable-bit-rate (VBR) video.**

## I. Introduction

FUTURE broad-band networks will support a wide variety of services with very different traffic characteristics. Among them, multimedia applications such as video-on-demand (VoD) are expected to consume a significant portion of the bandwidth. Therefore, the efficient transmission of delay-sensitive variable-bit-rate (VBR) video data [1] is likely to be one of the key challenges in managing resources in such networks. Apart from the frame-by-frame bit-rate fluctuations that are also found in constant-bit-rate (CBR) videos, VBR videos tend to exhibit long-range bit-rate variations on a time scale of minutes. Such fluctuations complicate the admission of video streams and the scheduling of video data transmission to provide performance guarantee.

To tackle this problem, researchers have studied extensively various ways to reduce the bit-rate variations of prerecorded VBR videos [2]–[9]. These include the use of a *smoothing buffer* located at the client [2]–[8] or at intermediate nodes over the network [9], to smooth out the video bit rates. The smoothing can then be optimized to minimize the number of rate increases [3], to minimize the number of rate changes [4], or to minimize the variability of the rates [8], and so on. Interested readers are referred to the excellent study by Feng and Rexford [10] for a detailed survey and comparison of various smoothing algorithms.

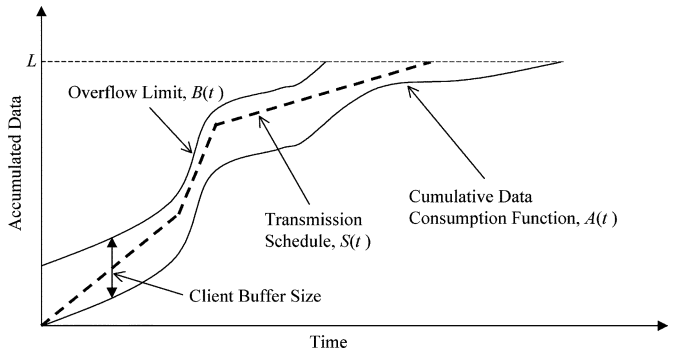



Fig. 1. Feasible piecewise-smooth transmission schedule for VBR video delivery.

After smoothing, the transmission schedule of a VBR video will be reduced into a series of constant-rate segments (see Fig. 1). The video server can then reserve bandwidth for these segments before transmitting them over the network to the client. As long as the bandwidth reservations are successful, timely delivery of the video data to the client can be guaranteed. However, two factors in practice often affect the effectiveness of this approach.

First, although the bit rate of each smoothed segment is constant, the system still needs to successfully complete the bandwidth reservation process before the next segment can be transmitted. The adjustments needed may contain both downward adjustments (switching from a higher bit rate to a lower bit rate), and upward adjustments (switching from a lower bit rate to a higher bit rate). The former case is straightforward as less network resources will be required, but the latter case is more complicated. In particular, if the network concurrently carries traffic from other applications (e.g., Web, FTP, or other video streams), it is conceivable that the upward adjustments could fail when the additional bandwidth is not available *at that moment*. Clearly this will result in either disruption of the video stream or severe quality degradation such as playback jitter. As the *instantaneous* bandwidth consumption in a network with mixed traffics is inherently unpredictable, this problem is unavoidable unless one dedicates a portion of the network resources to a video stream.

Manuscript received August 26, 2002; revised May 21, 2003. This work was supported in part by a Direct Grant and Earmarked Grants CUHK 4209/01E and CUHK 4328/02E from the HKSAR Research Grant Council and in part by the Area of Excellence Scheme, established under the University Grants Council of the Hong Kong Special Administrative Region, China, under Project AoE/E–01/99. This paper was recommended by Associate Editor H. Sun.

The authors are with the Department of Information Engineering, The Chinese University of Hong Kong, Shatin, N.T., Hong Kong (e-mail: hllai9@ie.cuhk.edu.hk; jacklee@computer.org; lkchen@ie.cuhk.edu.hk).

Digital Object Identifier 10.1109/TCSVT.2004.841687

However, this clearly will result in significant over-engineering and thus defeat the whole purpose of smoothing in the first place.

Second, a subtler problem with bandwidth adjustment is the processing delay. Regardless of the resource reservation protocols adopted, a sender (e.g., a video server) that desires to adjust a connection's bandwidth must first send protocol messages to one or more network controllers (e.g., routers). The network controllers may in turn need to contact other controllers along the path of the connection before the request can be granted or denied. In any case, this process will take time and the time it takes will depend on a lot of factors, such as the network topology, the reservation protocol adopted, the current utilization of the network, the number of resource-reservation requests being processed, and loss of control messages. The point is, not only the processing itself takes time, the time it takes also varies. This creates another problem in upward bit-rate adjustments as delay or even transmission losses may occur if an adjustment cannot be completed in time. Conceivably, one can issue the upward adjustments well ahead of time to prevent delay/loss, but estimating the correct lead-time is by no means trivial.

Motivated by the previous two problems, we investigate in this study a new scheduler for transmitting VBR videos that can provide deterministic performance guarantee even in a mixed-traffic network and is immune to the random delays in processing network resource reservation requests. The principle of the scheduler, called the monotonic decreasing rate (MDR) scheduler, is to eliminate upward bit-rate adjustments altogether. That is, the transmission schedule is composed of a series of segments, of which each segment is assigned a bit rate strictly lower than the previous segment. Now, without the need for upward bit-rate adjustment, resource reservations are guaranteed to be successful. Moreover, the timing of the bit-rate adjustments is no longer critical as video data transmission will not be affected by a later-than-expected downward bit-rate adjustment.

Intuitively, one will expect the MDR scheduler to require more client buffers as video data are transmitted more aggressively than other smoothing algorithms. Using real-world VBR video bit-rate traces, we quantify the tradeoff and show that, for some video streams, the buffer requirement is indeed increased when compared to smoothing algorithms. To tackle this problem, we propose an aggregated monotonic decreasing rate (AMDR) scheduler to enable one to control the buffer requirement to the same level as smoothing algorithms. Surprisingly, simulation results show that the AMDR scheduler can achieve performance comparable to existing smoothing algorithms even when equipped with the same buffer requirement. Thus, using the AMDR scheduler, one can provide performance guarantee in streaming VBR videos over mixed-traffic networks with no tradeoff in terms of admission complexity, network utilization, client waiting time, and client buffer requirement.

The rest of the paper is organized as follows. Section II introduces video bit-rate smoothing algorithms and reviews some previous works. Section III presents the MDR scheduler and analyzes its properties. Section IV evaluates performance of the MDR scheduler and compares it to optimal smoothing. Section V presents the AMDR scheduler and compares its performance with optimal smoothing. Section VI concludes the study.

## II. Video Bit-Rate Smoothing

In this section, we review some existing video bit-rate smoothing algorithms and define some notations that are used in the rest of the paper. Video bit-rate smoothing is a technique to reduce bit-rate variations in the retrieval and transmission of VBR encoded videos. The principle of smoothing is *work-ahead*, i.e., by transmitting data at a bit rate higher than the playback bit rate during periods of lower playback rates. Excess video data are then buffered at the client side so that the transmission bit rate can be reduced during periods of high playback rates by consuming video data from the buffer for playback. Note that smoothing not only reduces bit-rate variations, but also reduces the peak data rate as well.

Let $A(t)$ be the cumulative data consumption function for a video (see Fig. 1), defined as the amount of data that needs to be accumulated at the client for playback $t$ seconds after playback starts. Let $S(t)$ be the transmission schedule for the video, defined as the amount of data transmitted to the client $t$ seconds after playback starts. Ignoring network delay, processing delay, and interactive playback controls, it is clear that a feasible transmission schedule must not be lower than $A(t)$ for all $t$ so that the client will not run out of video data during playback:

$$S(t) \geq A(t). \tag{1}$$

On the other hand, if the client buffer size is limited to, say, $b$ bytes, then the transmission schedule cannot be too aggressive either or else client buffer overflow will occur. This buffer constraint can be represented by a function $B(t)$, defined as

$$B(t) = A(t) + b \tag{2}$$

and thus, to prevent buffer overflow, we must ensure that

$$S(t) \leq B(t). \tag{3}$$

Together, the two curves $A(t)$ and $B(t)$ define the feasible region for all feasible transmission schedules

$$B(t) \geq S(t) \geq A(t). \tag{4}$$

Clearly, there are an infinite number of feasible transmission schedules than can fit within the feasible region. Thus, one can pick a transmission schedule to optimize various measures of the system's performance. For example, McManus and Ross [2] suggested dividing the entire video stream into fixed size intervals and then transmitting each interval with a constant bit rate to enable control of the rate adjustment frequencies. Feng *et al.* investigated smoothing algorithms to minimize the number of rate increases [3] and to minimize the number of rate changes [4]. In another two studies by Feng [5], [6], he observed that, in some cases, an algorithm targeted at minimizing a certain parameter might make too aggressive prefetches or allow too large buffer residency times, so he proposed a rate-constrained smoothing algorithm [5] and a time-constrained smoothing algorithm [6] to solve these problems. Chang *et al.* suggested that transmitting at a constant rate yields lower overhead and complexity. Therefore, they proposed a smoothing algorithm that switches a single constant transmission rate on and off to adapt to the video

bit rates [7]. Salehi *et al.* investigated the optimal smoothing algorithm [8] that produces smoothing schedules with minimum peak rates and rate variations.

Besides smoothing algorithms based on finding a path inside the feasible region, there are also other related studies in this area. For example, Zhang [9] proposed smoothing using buffers located in multiple intermediate nodes in the network. Zhao and Tripathi [11] proposed an algorithm to multiplex smoothed VBR streams to further reduce bit-rate variations. Liu *et al.* [12] observed that scene changes in a video usually correlates with bit-rate variations and thus proposed an algorithm to detect scene changes to allocate a constant bit rate for each scene. For real-time videos, Rexford *et al.* [13] proposed an online, lossless smoothing algorithm that works with a sliding window. Liew and Tse [14] proposed using client buffer occupancy to control encoding parameters for smoother encoder output. In another study, Duffield *et al.* [15] used network status feedback to control the encoding parameters.

## III. MDR Scheduler

As discussed in Section I, the fundamental limitation of existing smoothing algorithms is the need for upward rate adjustments, of which correct operation depends on the successful and timely completion of network resource reservations. To remove this limitation, we propose using only downward rate adjustments in the transmission schedule. In other words, the initial transmission rate will be the highest, with each subsequent rate lower than the previous one. We call this algorithm monotonic decreasing rate (MDR) scheduler for obvious reasons.

In this study, we focus on prerecorded videos. The MDR schedule for a video is computed offline and is stored with the video for use during video streaming. We present an algorithm to compute the MDR schedule in the next section and derive several properties of the algorithm in Sections III-B–III-D.

### A. Computing the MDR Schedules

Compared to existing smoothing algorithms, the MDR property introduces an additional constraint—only downward rate adjustments can be used. Note that, although this reduces the set of possible schedules within the feasible region, it still does not uniquely determine the transmission schedule for a given video. In fact, there are still an infinite number of possible MDR schedules within the feasible region.

To select an MDR schedule, we need to consider the resultant resource requirements. The choice of the MDR schedule can affect the peak transmission rate and the client buffer requirement, both of which should be minimized. Interestingly, it turns out that we can always compute an MDR schedule that has minimum peak rate *and* minimum client buffer requirement, among all possible MDR schedules.

We define an MDR transmission schedule with the set of rate-time tuples: $\{r_i, T_i \mid i = 1, 2, \ldots, n\}$, where $r_i$ and $T_i$ are the transmission rate and commencing time for the $i$th segment in the transmission schedule, as depicted in Fig. 2, and $n$ is the total number of segments in the MDR transmission schedule. For a MDR transmission schedule, the rates will be monotonic decreasing, i.e., $r_i > r_j$, for all $i, j$, where $n \geq j > i \geq 1$.

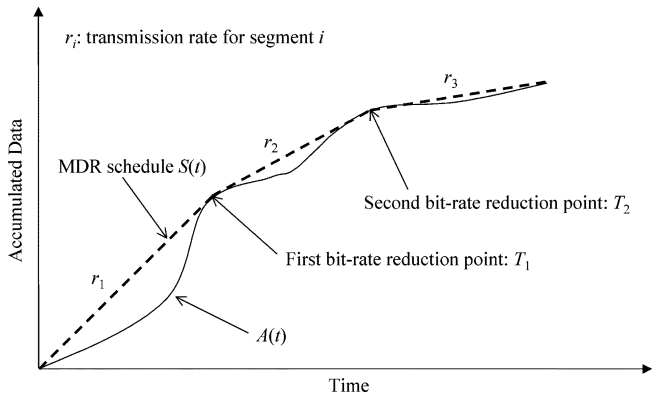



Fig. 2. Monotonic decreasing rate schedule generated by the MDR scheduler.

To compute the schedule, we begin from the origin as depicted in Fig. 2 and assign the first segment with the highest transmission rate, i.e.,

$$r_1 = \max\left\{ \frac{A(t) - A(0)}{t} \middle| \forall t > 0 \right\} \tag{5}$$

and mark the time, denoted by $T_1$, at which the rate is maximized. The tuple $\{r_1, T_1\}$ then represents the first segment of the MDR transmission schedule. Next, we repeat this process with $T_1$ as the starting point to obtain $\{r_2, T_2\}$, and so on. In general, the transmission rate for the next segment can be computed from

$$r_{i+1} = \max\left\{ \frac{A(t) - A(T_i)}{t - T_i} \middle| \forall t > T_i \right\} \tag{6}$$

until it reaches the end of the video. It can be shown (see Appendix A) that the above procedure guarantees that the generated transmission schedules are monotonic decreasing. The transmission schedule is then defined from the resultant rate-time tuples $\{r_i, T_i\}$ as follows:

$$S(t) = \int_0^t s(t), \text{ where } s(t) = r_i, \qquad \text{for } T_i \leq t < T_{i+1}. \tag{7}$$

The rate-time tuples are then stored together with the video data. The video server will simply schedule the transmission of the video according to this MDR transmission schedule. Its monotonicity property ensures that, once a stream is admitted, there will always be sufficient system bandwidth for the whole duration of the video stream, even if there are other random traffic such as web or file transfer in the system.

This MDR scheduler has several additional desirable properties, namely modest admission complexity, minimum peak rate, and minimum client buffer requirement, that are discussed in the following sections.

### B. Admission Complexity

We first consider admission complexity, defined as the number of computations needed to determine if a new video stream can be admitted to a system with finite bandwidth. For existing smoothing algorithms with transmission schedules consisting of both upward and downward rate adjustments, it is necessary to check the system's bandwidth availability to

determine if admitting the new stream will exceed the system capacity.

Let $U$ be the total system capacity and $U(t)$ be the system utilization at time $t$. Suppose that the new stream request arrives at time $t_0$; then the system can admit the new stream if and only if there is sufficient system capacity available for the entire duration of the new video stream, i.e.,

$$U(t) + S(t - t_0) \leq U, \quad \text{for all } t \text{ from } t_0 \text{ to } (t_0 + L). \quad (8)$$

In practice, we do not compute (8) in the continuous time domain as I/O schedules are likely to be organized into service rounds (e.g., disk retrieval rounds). Let $\delta$ be the round length. Then, for a video of length $L$, there will be $w = L/\delta$ rounds. For clarity, we refer to the system scheduler's cycles as rounds, counting from zero from the startup of the system, and refer to a video title's data unit to be retrieved and transmitted in a round as a block, counting from zero from the beginning of the video.

Let $u_i$ $(i = 0, 1, \ldots)$ be the system's utilization in round $i$, and $v_j$ $(j = 0, 1, \ldots, w-1)$ be the transmission rate of block $j$ of the video, which can be computed from the transmission schedule $\{r_i, T_i\}$. Then, for a new client arriving at round $A$, the admission process will require $w$ additions to compute the new aggregate bandwidth utilization for round $A$ to round $A+w-1$

$$u_i = u_i + v_{i-A}, \quad i = A, A+1, \ldots, A+w-1 \quad (9)$$

and $w$ comparisons to determine if the network capacity is exceeded in any of those $w$ rounds. The admission complexity is then of order $\mathrm{O}(2w)$ for a successful admission.

For an unsuccessful admission, the complexity will be lower as the admission test in (9) can be stopped once the system utilization is exceeded in any of the $w$ rounds. In this case, the client will have to wait until the next round to repeat the admission test. This process repeats until either the client is admitted or the client leaves the system due to excessive wait. Therefore, the admission complexity is further multiplied by the waiting time.

By contrast, the MDR scheduler requires a very simple admission test with only one single computation. As MDR schedules are all monotonic decreasing, this implies that the available system bandwidth utilization $u_i$ is a nonincreasing series. It follows that, if the first transmission rate can be accommodated, i.e.,

$$u_A + v_0 \leq U \quad (10)$$

then the rest of the schedule is guaranteed to not exceed the system capacity as well:

$$\begin{aligned} & u_i + v_{i-A}, \quad i = A, A+1, \ldots, A+w-1 \\ & \leq u_A + v_{i-A}, \quad \because u_i \text{ is nonincreasing} \\ & \leq u_A + v_0, \quad \because v_j (j = 0, 1, \ldots, w-1) \text{ is nonincreasing} \\ & = U, \quad \because (10). \end{aligned} \quad (11)$$

If the admission is successful, then the system utilization $u_i$'s will be updated according to (9). Otherwise, the admission test will be repeated in the next round. There is one key difference compared to general smoothing algorithms: the system utilization update needs to be performed once only, even if the client has to wait and perform multiple rounds of admission tests. This is because the admission test can be completed using (10) without computing (9) under the MDR scheduler. This leads to significantly lower complexity in the admission process.

### C. Peak Transmission Rate

The first rate in a MDR schedule is the peak transmission rate. Compared to the video's data consumption function $A(t)$, this peak rate $r_1$ is bounded from the above by the consumption function's peak rate

$$r_0 \leq \max \left\{ \frac{dA(t)}{dt} | \forall t \geq 0 \right\}. \quad (12)$$

The equality will only occur when the peak rate of the cumulative data consumption function appears right at the origin. In this case, any feasible transmission schedule with zero startup delay will have a startup rate larger than or equal to the startup rate of the cumulative data consumption function. This is stated in Theorem 1 below.

*Theorem 1: The MDR scheduler generates schedules with the minimum peak rate among all feasible schedules with zero startup delay.*

*Proof:* The peak rate of an MDR schedule is the first rate $r_1$, so it is sufficient to prove that $S(t)$ has the smallest slope among all feasible schedules for $t$ in $0 \leq t < T_1$. We prove this by contradiction. Let $Y(t)$ be a feasible schedule with zero startup delay

$$Y(t) \geq A(t) \quad (13)$$

and has bit rate no larger than the MDR schedule $S(t)$ before the first bit-rate reduction point $T_1$, i.e.,

$$Y'(t) \leq S'(t), \quad \text{for } 0 \leq t < T_1. \quad (14)$$

As $S(t)$ is a straight line in the range $(0, T_1)$, equality in (14) holds only if $Y(t)$ is equivalent to $S(t)$. As we assumed they are different, that implies

$$Y'(t) < S'(t), \quad \text{for } 0 \leq t < T_1. \quad (15)$$

Integrating (14) on both sides with respect to $t$ yields

$$\int_0^{T_1} Y'(t)dt < \int_0^{T_1} S'(t)dt \quad (16)$$

and we obtain

$$\begin{aligned} Y(T_1) &< S(T_1) \\ &= A(T_1) \end{aligned} \quad (17)$$

which implies that there will be a buffer underflow at the point $T_1$. This contradicts our assumption that $Y(t)$ is a feasible schedule and thus the result follows. ■

### D. Client Buffer Requirement

Similar to video smoothing, the MDR scheduler also requires the client to buffer video data ahead of their playback schedule.

Given an MDR schedule $S(t)$, the buffer requirement is the maximum difference between the transmission curve $S(t)$ and the data consumption curve $A(t)$

$$B = \max\{S(t) - A(t) \mid \forall t \geq 0\}. \tag{18}$$

As discussed in Section III-A, there are infinitely many feasible transmission schedules that are also monotonic decreasing. The one defined in Section III-A, however, has the minimum client buffer requirement as stated in Theorem 2 below.

*Theorem 2: The MDR scheduler generates schedules with the minimum buffer requirement among all feasible monotonic decreasing rate schedules.*

*Proof:* We will prove by contradiction. Let $X(t)$ be a feasible monotonic decreasing rate schedule, i.e.,

$$X(t) \geq A(t). \tag{19}$$

Assume that $X(t)$ has a lower buffer requirement than the MDR transmission schedule $S(t)$, then

$$\exists t_0 \text{ such that } S(t_0) > X(t_0) \geq A(t_0). \tag{20}$$

We know that $S(t)$ must coincide with $A(t)$ at the bit-rate reduction points, i.e.,

$$S(T_i) = A(T_i), \quad \text{for } i = 1, 2, \ldots, n \tag{21}$$

where $n$ is the number of bit-rate reduction points. Now $X(t)$ cannot be lower than $S(t)$ at the bit-rate reduction points $\{T_i \mid i = 0, 1, 2 \ldots n\}$. This implies that the $t_0$ in (20) cannot be the bit-rate reduction points

$$t_0 \neq T_i, \quad \text{for } i = 1, 2, \ldots, n. \tag{22}$$

However, as $S(t)$ is constructed with straight lines connecting the bit-rate reduction points, $X(t)$ cannot be lower than $S(t)$ in between two consecutive bit-rate reduction points either

$$t_0 \notin (T_{i-1}, T_i), \quad \text{for } i = 2, 3, \ldots, n. \tag{23}$$

Otherwise, $X(t)$ will be convex in the range $(T_{i-1}, T_i)$, which contradicts with the assumption that $X(t)$ has monotonic decreasing rates. From (21) and (22), we conclude that $t_0$ does not exist and the result follows. ∎

## IV. Performance Evaluation

We evaluate performance of the MDR scheduler and compare it to optimal smoothing [1] in this section. To obtain realistic performance results, we collected the video bit-rate traces of 274 different videos from DVD movies for simulation. These are full-length (average 5781 s long and 4348 MB in size), MPEG-2 encoded videos with an average bit rate of 6.02 Mb/s. The bit rate varies from below 0.5 to over 18 Mb/s. Long-range (minutes to tens of minutes) bit-rate variations are common in these real-world MPEG-2 encoded videos.

We implemented the MDR scheduler presented in Section III-A in software and use it to compute the transmission schedule for the videos. We also implemented the optimal smoothing algorithm [1] for comparison purposes. The generated transmission schedules are then fed into a simulator developed using CNCL [16] to obtain simulation results.

The simulation model consists of a system with clients connecting through a 1-Gb/s backbone network to a server storing the 274 VBR videos. We assume that the backbone network is the bottleneck of the system. For simplicity, we ignore delay and loss in the network. New stream requests are generated according to a Poisson process with various mean interarrival times to simulate different system utilization. A new stream request randomly selects a video from the 274-video collection with uniform probability. Note that we adopt the uniform popularity instead of the Zipf popularity model [17], because the video titles have varying bandwidth requirements and lengths. Consequently, using the Zipf popularity will result in large variations in the simulation results, depending on which of the video titles happened to be picked as the hot titles. To obtain more consistent results for comparison, we therefore adopt the uniform popularity model.

To admit a new stream, network resource reservations are performed according to the generated transmission schedules on a per-stream basis. The admission test for schedules generated by optimal smoothing is performed with a round length of one second (see Section III-B). Each simulation run simulates a duration of 3000 days. We summarize the results in the following sections.

### A. Admission Complexity

Table I compares the admission complexity of optimal smoothing and the MDR scheduler. The simulation results are obtained from counting the average number of computations required to admit a new client. We separate the computations incurred in unsuccessful and successful admissions. For unsuccessful admissions, the computation complexity is comparable for the MDR scheduler and the optimal smoothing scheduler. By contrast, the MDR scheduler requires significantly fewer computations than the optimal smoothing scheduler for successful admissions, which dominates the total admission complexity.

### B. Waiting Time Versus System Utilization

To evaluate the bandwidth efficiency of the MDR scheduler, we collected the mean and worst-case client waiting times for both schedulers and plot the results in Fig. 3 for three system bandwidth settings. The results show that the MDR scheduler achieves performance similar to optimal smoothing for all three system bandwidth settings and across system utilization from 10% to 90%. This suggests that the MDR scheduler can guarantee VBR video delivery in mixed-traffic networks with negligible tradeoff in latency—the key performance metric experienced by the end users.

### C. Client Buffer Requirement

Although Theorem 2 shows that the generated MDR schedule always has the minimum buffer requirement among all monotonic decreasing rate schedules, the actual buffer requirement

TABLE I
COMPARISON OF ADMISSION COMPLEXITY BETWEEN THE OPTIMAL SMOOTHING AND THE MDR SCHEDULERS

| Scheduler | Unsuccessful Admission # Comparisons | Unsuccessful Admission # Additions | Successful Admission Comparisons | Successful Admission Additions |
|---|---|---|---|---|
| Temporal smoothing | | | | |
| - complexity at $\rho$=0.6* | 0.00 (0.00) | 0.00 (0.00) | 5782.52 | 5782.52 |
| - complexity at $\rho$=0.7* | 0.00 (0.00) | 0.00 (0.00) | 5782.20 | 5782.20 |
| - complexity at $\rho$=0.8* | 0.19 (0.17) | 0.19 (0.17) | 5782.04 | 5782.04 |
| - complexity at $\rho$=0.9* | 5.32 (4.89) | 5.32 (4.89) | 5782.08 | 5782.08 |
| MDR scheduler | | | | |
| - complexity at $\rho$=0.6* | 0.00 (0.00) | 0.00 (0.00) | 1 | 5782.60 |
| - complexity at $\rho$=0.7* | 0.00 (0.00) | 0.00 (0.00) | 1 | 5782.28 |
| - complexity at $\rho$=0.8* | 0.17 (0.17) | 0.17 (0.17) | 1 | 5782.11 |
| - complexity at $\rho$=0.9* | 4.83 (4.83) | 4.83 (4.83) | 1 | 5782.16 |

* Numerical results are measured as the average number of computations required to admit a client at a given average network utilization ($\rho$), averaged over requests for all 274 videos.

# Numbers in parenthesis are the average number of unsuccessful admission tests performed for each request.

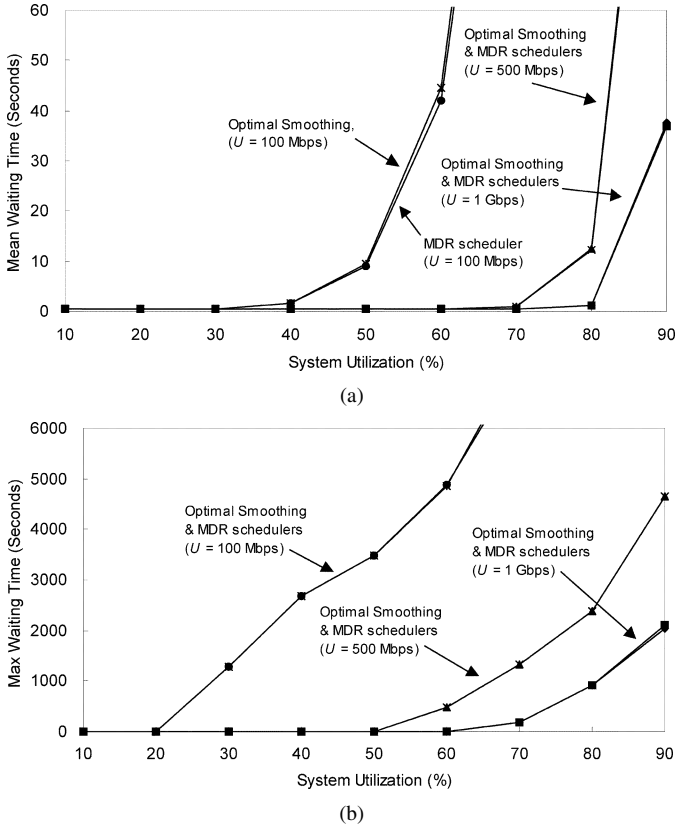



Fig. 3. (a) Comparison of mean waiting time versus system utilization for the optimal smoothing and the MDR schedulers (system bandwidth is equal to $U$). (b) Comparison of worst-case waiting time versus system utilization for the optimal smoothing and the MDR schedulers (system bandwidth is equal to $U$).

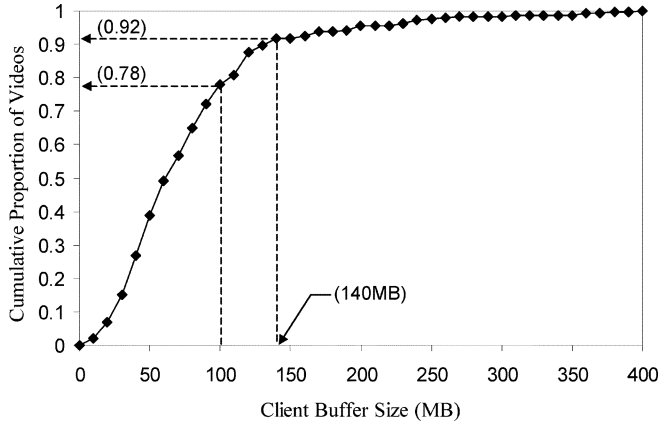



Fig. 4. Distribution of client buffer requirement for MDR transmission schedules.

still depends on $A(t)$ or the bit-rate profile of the video. Fig. 4 shows the distribution of client buffer requirements for the 274 videos tested. The average amount of client buffer required is 76.4 MB. Out of the 274 tested videos, 78% require no more than 100 MB of client buffer and 92% require no more than 140 MB of client buffer. The worst-case client buffer requirement is 394.5 MB. This result is strikingly close to the 20/80 rule, also known as the Pareto's principle—78% of videos require no more than 20% (100 MB/394.5 MB) of client buffer.

With the trend toward integrating multiple information and entertainment services ranging from the Web and network gaming to a digital video recorder into a home entertainment center device, the added buffer requirement can easily be accommodated. Nevertheless, the 20/80 observation does suggest that the client buffer utilization will be low most of the time. We investigate in the next section an alternative solution that provides better control of the buffer requirement.

## V. AGGREGATED MONOTONIC DECREASING RATE SCHEDULER

Results from the previous section show that the MDR scheduler can achieve performance comparable to optimal smoothing and yet can provide guaranteed video delivery in a mixed-traffic network. The tradeoff, as Section IV-C reveals, is the increased client buffer requirement, which in a few rare cases reaches close to 400 MB. While even this amount of client buffer can easily be accommodated in future home entertainment centers with a built-in hard disk, there are still two inefficiencies.

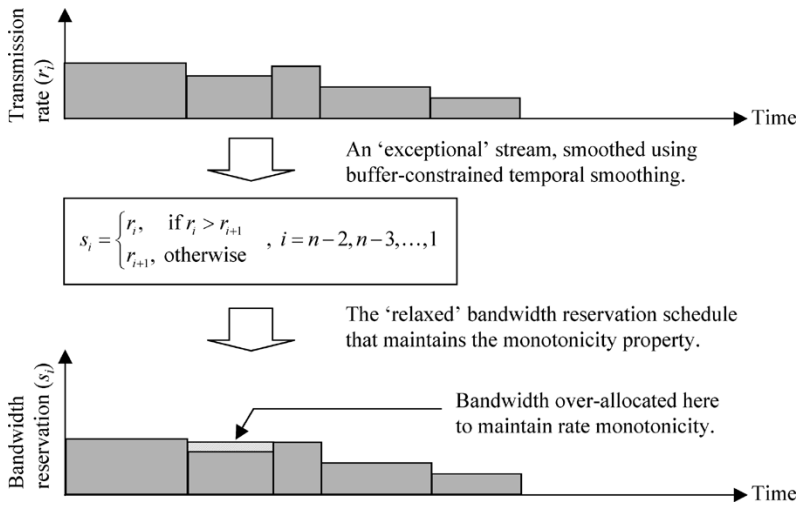



Fig. 5. Preserving the monotonicity property by overallocating bandwidth.

First, our results show that 78% of the video titles in our collection of 274 full-length video titles require no more than 20% of the worst-case buffer requirement. This suggests that the client buffer utilization will be low most of the time and most of the reserved buffer space will be unused.

Second, although video titles with exceedingly large client buffer requirements are rare, the MDR scheduler cannot prevent such cases as the buffer requirement depends on the individual video's bit-rate profile. In other words, the worst-case buffer requirement is, in theory, unbounded.

To tackle these two deficiencies, we introduce in this section an aggregated monotonic decreasing rate (AMDR) scheduler that applies the MDR principle to aggregated network flows so that relaxed transmission schedules can be used to accommodate those rare video titles that otherwise require very large buffer requirements.

### A. Bandwidth Reservation

The large buffer requirement in those rare videos is a result of the MDR scheduler's monotonicity property. However, as we prove in Section III-D, the MDR scheduler already achieves the minimum buffer requirement among all monotonic decreasing rate schedules, implying that the only way to reduce the buffer requirement is to relax the monotonicity requirement.

Under the AMDR scheduler, the client buffer requirement, say $B$, is specified by the service provider as a design parameter. For videos with buffer requirements smaller than or equal to $B$, they are delivered using the original MDR schedules. By contrast, for videos that have buffer requirements larger than $B$, they are delivered using buffer-constrained schedules generated using temporal smoothing algorithms such as optimal smoothing.

Now, obviously temporal smoothing in general does not guarantee monotonicity and this implies that we can no longer guarantee video delivery for these videos in a mixed-traffic network. To tackle this problem, we can overallocate bandwidth for these video streams such that the *reserved bandwidth allocations* are kept monotonic decreasing. For example, consider the transmission schedule $\{r_i, T_i \mid i = 1, 2, \ldots, n\}$ generated by a temporal smoothing algorithm with buffer constraint $B$. For the rare videos, there exists $i$, $j$, such that $r_i < r_j$ for $i < j$. We can maintain the monotonicity property by applying the following procedure to the transmission schedule:

$$s_i = \begin{cases} r_i, & \text{if } r_i > r_{i+1} \\ r_{i+1}, & \text{otherwise} \end{cases}, i = n-2, n-3, \ldots, 1 \quad (24)$$

to obtain the bandwidth reservation schedule $\{s_i, T_i \mid i = 1, 2, \ldots, n\}$, as shown in Fig. 5. Note that the transmission schedule is not changed, only the bandwidth reservations are modified to maintain the monotonicity property, albeit at the expense of some unused network bandwidth.

To reduce the inefficiency due to bandwidth overallocation, we observe that a video server often serves many video streams simultaneously. The data for these video streams typically go through the same backbone network before reaching the access networks. For a network link carrying more than one video stream, if we can ensure that the aggregate traffic conforms to the monotonicity property, the delivery of the individual streams is also guaranteed, even with mixed network traffics. In this case, we are applying the MDR principle to the aggregate traffic flow instead of individual video streams.

When a video stream with transmission schedule generated by the MDR scheduler is admitted to the system, say at round $A$, we simply add the transmission schedule to the aggregate bandwidth utilization to obtain the new system utilization

$$u_i = u_i + v_{i-A}, \quad i = A, A+1, \ldots, A+w-1. \quad (25)$$

The bandwidth reservation schedule will then be set equal to the system utilization, i.e., $s_i = u_i$.

On the other hand, when a video stream with transmission schedule generated by the optimal smoothing algorithm is admitted to the system at round $A$, we will need to perform an additional step to maintain monotonicity for the aggregated bandwidth reservations. We first compute the system utilization using

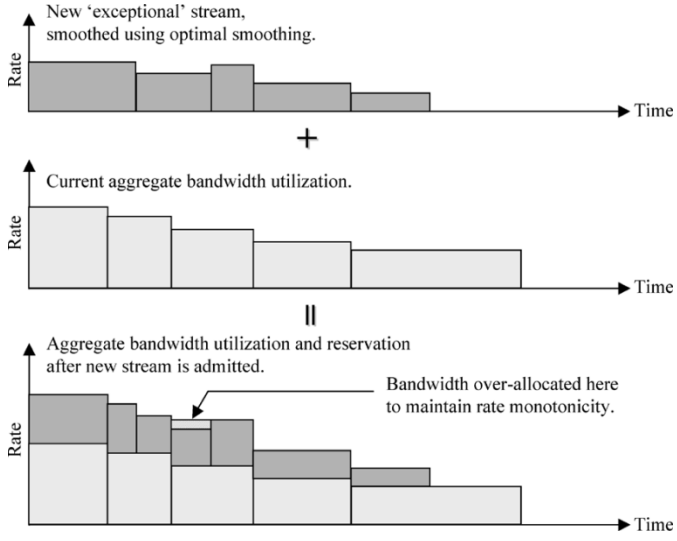



Fig. 6. Preserving the monotonicity property in the AMDR scheduler.

(25). Then we apply a procedure similar to (24) to compute an MDR bandwidth reservation schedule by overallocations

$$s_i = \begin{cases} u_i, & \text{if } u_i > u_{i+1} \\ u_{i+1}, & \text{otherwise} \end{cases}, i = A+w-2, A+w-3, \ldots, A \tag{26}$$

as shown in Fig. 6. Again, the bandwidth overallocations only affect the amount of network resources reserved. The individual video stream's transmission schedule is not affected.

### B. Admission Complexity

The admission complexity of the AMDR scheduler depends on whether the requested video is delivered using an MDR transmission schedule or an Optimal Smoothing transmission schedule. For the MDR case, the admission complexity is the same as in the original MDR scheduler, i.e., one computation for the admission test and $\mathrm{O}(w)$ computations for updating the system utilization series.

For the optimal smoothing case, the admission complexity is higher than the MDR case but, interestingly, lower than the original optimal smoothing case. This is because, in the AMDR scheduler, the bit rates in the bandwidth reservation schedule $\{s_i\}$ are nonincreasing. This enables the system to perform the admission test by checking only the initial rate and the rate-increasing rounds.

Again assuming that the client arrives at time slot $A$, with a transmission schedule $\{v_i\}$, we define a round $i$ in the transmission schedule as rate increasing if $v_i > v_{i-1}$. Let there be $g$ such rate-increasing rounds, with the round number denoted by $h_i, i = 1, 2, \ldots, g$. To simplify notations, we also define $h_0 = 0$ to represent the initial round. With these notations, we can then define the admission test as

$$s_{h_i+A} + v_{h_i} \leq U, \quad \text{for } i = 0, 1, \ldots, g. \tag{27}$$

The monotonicity property of the bandwidth reservation schedule implies that

$$s_{i+A} + v_i \leq U \Rightarrow s_{i+A+1} + v_{i+1} \leq U. \tag{28}$$

Starting with $i = h_0, h_1, \ldots, h_g$ and then applying (28) recursively, we can show that, if a new stream's transmission schedule satisfies (27), the whole transmission schedule can be added to the bandwidth reservation schedule without exceeding the system capacity.

The admission test therefore requires $(g+1)$ additions and comparisons, instead of the $w$ ($w \gg g$) additions and comparisons in the original temporal smoothing case. Once the admission test is successful, then the new stream's transmission schedule will be added to the aggregate system utilization using (25), and then the system can compute the new bandwidth reservation schedule according to (26).

Assume that a proportion of $\alpha$ ($0 \leq \alpha \leq 1$) of the video collection can be admitted using MDR schedules under a given client buffer size constraint. Then, for successful admissions, the admission complexity is equal to $\mathrm{O}(1 + (1-\alpha)(g+w))$ comparisons and $\mathrm{O}(w)$ additions. For unsuccessful admissions, again the complexity will be lower as the admission test is stopped as soon as the system utilization is exceeded in a time slot.

### C. Performance Evaluation

We evaluate the AMDR scheduler's performance in this section using simulation. The simulation setup is identical to the one in Section IV except for two differences: 1) the MDR scheduler is replaced by the AMDR scheduler and 2) the client buffer size is fixed.

Table II shows the admission complexity for the AMDR scheduler and the optimal smoothing scheduler. Comparing the results with those in Table I, we observe that the AMDR scheduler requires more computations than the MDR scheduler. This is because we have set a client buffer size constraint of 32 M for the AMDR scheduler, and, consequently, a proportion of the videos are scheduled using the optimal smoothing scheduler, which requires more admission computations. Nevertheless, the resultant admission complexity is still less than the optimal smoothing scheduler for both successful and unsuccessful admissions.

Next we investigate the impact on the client waiting time. Fig. 7 plots the mean and worst-case client waiting times versus client buffer size for a system utilization of 90%. We also simulated lower system utilization settings of 60%–80%, but both schedulers perform nearly identically and so the results are not shown here. From Fig. 7, we observe that, with smaller buffer sizes, both AMDR and optimal smoothing achieve similar waiting times. At larger buffer sizes, AMDR slightly outperforms optimal smoothing and ultimately converges to the MDR curve that has no buffer size constraint. The performance difference between AMDR and optimal smoothing is due to the fact that MDR schedules are more aggressive at the beginning of the video stream, where the transmission bit rate is highest. This results in more work-ahead as compared to optimal smoothing, and thus the MDR scheduler is able to utilize any unused bandwidth to reduce the bit-rate requirements down the road.

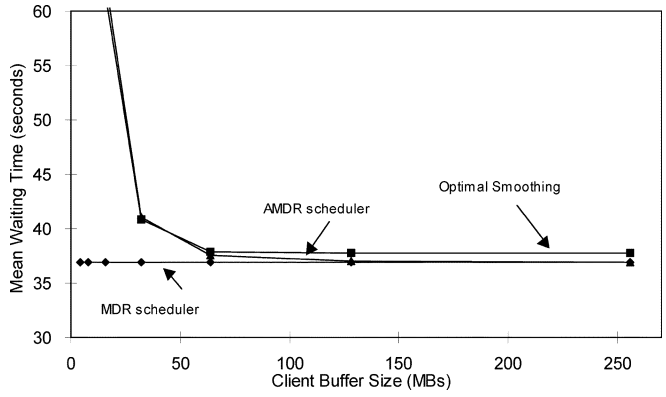
As the AMDR scheduler overallocates bandwidth to maintain a MDR schedule, it may become less efficient when the system

TABLE II
COMPARISON OF ADMISSION COMPLEXITY BETWEEN THE OPTIMAL SMOOTHING AND THE AMDR SCHEDULERS (CLIENT BUFFER SIZE FIXED AT 32 MB)

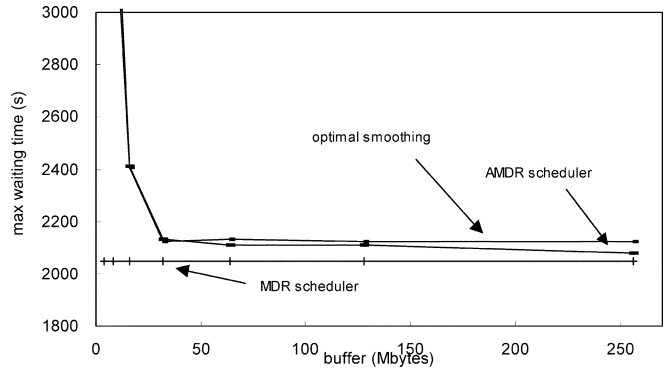
| Scheduler | Unsuccessful Admission # Comparisons | Unsuccessful Admission # Additions | Successful Admission Comparisons | Successful Admission Additions |
|---|---|---|---|---|
| Temporal smoothing | | | | |
| - complexity at $\rho$=0.6* | 0.00 (0.00) | 0.00 (0.00) | 5782.52 | 5782.52 |
| - complexity at $\rho$=0.7* | 0.01 (0.00) | 0.01 (0.00) | 5782.20 | 5782.20 |
| - complexity at $\rho$=0.8* | 4.15 (0.20) | 4.15 (0.20) | 5782.04 | 5782.04 |
| - complexity at $\rho$=0.9* | 74.00 (5.15) | 74.00 (5.15) | 5782.08 | 5782.08 |
| AMDR scheduler | | | | |
| - complexity at $\rho$=0.6* | 0.00 (0.00) | 0.00 (0.00) | 5008.15 | 5781.70 |
| - complexity at $\rho$=0.7* | 0.00 (0.00) | 0.00 (0.00) | 5007.47 | 5781.38 |
| - complexity at $\rho$=0.8* | 0.20 (0.20) | 0.20 (0.20) | 5007.15 | 5781.22 |
| - complexity at $\rho$=0.9* | 5.19 (5.17) | 5.19 (5.17) | 5007.08 | 5781.26 |

* Numerical results are measured as the average number of computations required to admit a client at a given average network utilization ($\rho$), averaged over requests for all 274 videos.

# Numbers in parenthesis are the average number of unsuccessful admission tests performed for each request.

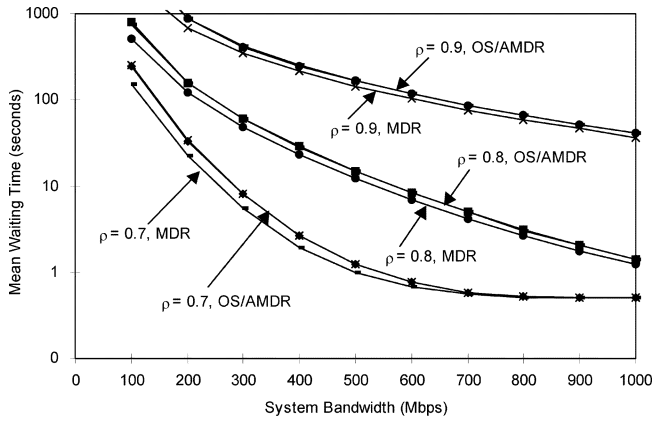



(a)

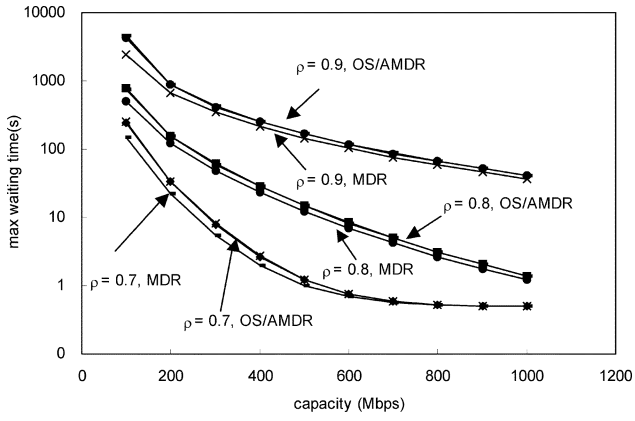



(b)

Fig. 7. (a) Mean client waiting time versus client buffer size. (b) Worst-case client waiting time versus client buffer size. (Note that the MDR scheduler does not have a buffer constraint).




(a)




(b)

Fig. 8. (a) Mean client waiting time versus system capacity. (b) Worst-case client waiting time versus system capacity.

capacity is small. To investigate this issue, we repeat the simulations for a range of system capacity from 100 Mb/s to 1 Gb/s and plot the mean and worst-case waiting times in Fig. 8. Comparing different schedulers, we observe that AMDR and optimal smoothing have nearly identical performance while the MDR scheduler consistently achieves lower waiting time. This is expected, as the MDR scheduler is not subject to buffer size constraint, which in this case equals 32 MB. Comparing the waiting time against the system bandwidth, we can see that the waiting time increases significantly at lower system bandwidth settings. Nevertheless, the differences between the AMDR scheduler and the optimal smoothing scheduler are negligible even for extremely small system bandwidth (e.g., 100 Mb/s). This shows that the overhead incurred in maintaining an MDR schedule in the AMDR scheduler is negligible.

## VI. Conclusion

By scheduling the transmission of video data in a monotonic-decreasing manner, we are able to deliver VBR videos in a mixed-traffic network with a deterministic performance guarantee. This enables the service provider to exploit the available bandwidth to support other nondelay-sensitive data services and thus improve network utilization. Through extensive simulations using 274 real-world VBR video bit-rate traces, the MDR is shown to achieve good performance in terms of waiting time under the same network utilization, comparable to that of optimal smoothing, while still being able to guarantee video delivery. For applications that require a bounded client buffer requirement, the proposed AMDR scheduler can be applied, and results show that the performance is nearly identical to optimal smoothing even for a buffer size as small as 32 MB. Thus, using the AMDR scheduler, one can provide performance guarantee in streaming VBR videos over mixed-traffic networks with no tradeoff in terms of admission complexity, network utilization, client waiting time, and client buffer requirement.

## Appendix
## Proof of MDR Scheduler's Monotonicity Property

*Theorem 3: Transmission schedules generated by the MDR scheduler are guaranteed to comprise monotonic decreasing rates.*

*Proof:* We prove the theorem by contradiction. Let $r_i$ and $r_{i+1}$ be the transmission rate of the $i$th and $(i+1)$th segments of a feasible schedule $S(t)$ generated by the MDR scheduler. Graphically, let $r'$ be the slope of the line connecting $S(T_{i-1})$ and $S(T_{i+1})$, where $T_{i-1}$ and $T_{i+1}$ are the $(i-1)$th and $(i+1)$th rate reduction points.

Assume that $r_i < r_{i+1}$, i.e., the rates allocated are not monotonic decreasing. Then we have

$$\frac{S(T_i) - S(T_{i-1})}{T_i - T_{i-1}} < \frac{S(T_{i+1}) - S(T_i)}{T_{i+1} - T_i} \tag{A1}$$

or

$$[S(T_i) - S(T_{i-1})](T_{i+1} - T_i) < [S(T_{i+1}) - S(T_i)](T_i - T_{i-1}). \tag{A2}$$

We expand (A2) to obtain

$$\begin{aligned} &S(T_i)T_{i+1} - S(T_i)T_i - S(T_{i-1})T_{i+1} + S(T_{i-1})T_i \\ &\quad < S(T_{i+1})T_i - S(T_{i+1})T_{i-1} - S(T_i)T_i + S(T_i)T_{i-1}. \end{aligned} \tag{A3}$$

We cancel the $S(T_i)T_i$ term on both sides and, after rearranging, we obtain

$$\begin{aligned} &S(T_i)T_{i+1} - S(T_{i-1})T_{i+1} - S(T_i)T_{i-1} \\ &\quad < S(T_{i+1})T_i - S(T_{i+1})T_{i-1} - S(T_{i-1})T_i. \end{aligned} \tag{A4}$$

Adding $S(T_{i-1})T_{i-1}$ to both sides, we obtain

$$\begin{aligned} &S(T_i)T_{i+1} - S(T_{i-1})T_{i+1} - S(T_i)T_{i-1} - S(T_{i-1})T_{i-1} \\ &\quad < S(T_{i+1})T_i - S(T_{i+1})T_{i-1} - S(T_{i-1})T_i - S(T_{i-1})T_{i-1}. \end{aligned} \tag{A5}$$

Then we factorize (A5) to get

$$\begin{aligned} &[S(T_i) - S(T_{i-1})](T_{i+1} - T_{i-1}) \\ &\quad < [S(T_{i+1}) - S(T_{i-1})](T_i - T_{i-1}) \end{aligned} \tag{A6}$$

which is equivalent to

$$\frac{S(T_i) - S(T_{i-1})}{T_i - T_{i-1}} < \frac{S(T_{i+1}) - S(T_{i-1})}{T_{i+1} - T_{i-1}} = r'. \tag{A7}$$

Since the transmission schedule coincides with the data consumption curve at bit-rate reduction points, i.e., $S(T_i) = A(T_i)$ for all $i$, we have

$$\frac{A(T_i) - A(T_{i-1})}{T_i - T_{i-1}} < \frac{A(T_{i+1}) - A(T_{i-1})}{T_{i+1} - T_{i-1}}. \tag{A8}$$

Now according to the MDR scheduler [see (6)], we have

$$r_i = \max\left\{ \frac{A(t) - A(T_{i-1})}{t - T_{i-1}} \middle| \forall t > T_{i-1} \right\}. \tag{A9}$$

From (A8), we must have $r_i$ equal to

$$r_i = \frac{A(T_{i+1}) - A(T_{i-1})}{T_{i+1} - T_{i-1}} \tag{A10}$$

which violates the definition of $r_i$. This contradicts with the assumption that the schedule $S(t)$ is generated by the MDR scheduler and therefore the schedule with increasing rates cannot be generated by the MDR scheduler, and the result follows. ■

## Acknowledgment

The authors want to express their gratitude to the anonymous reviewers for their insightful comments and suggestions in improving this paper.

## References

[1] T. V. Lakshman, A. Ortega, and A. R. Reibman, "VBR video: Tradeoffs and potentials," *Proc. IEEE*, vol. 86, no. 5, pp. 952–973, May 1998.

[2] J. McManus and K. Ross, "Video on demand over ATM: Constant-rate transmission and transport," *Proc. IEEE INFOCOM*, pp. 1357–1362, Mar. 1996.

[3] W. Feng and S. Sechrest, "Critical bandwidth allocation for the delivery of compressed video," *Comput. Commun.*, vol. 18, no. 10, pp. 709–717, Oct. 1995.

[4] W. Feng, F. Jahanian, and S. Sechrest, "Optimal buffering for the delivery of compressed prerecorded video," *ACM Multimedia Syst. J.*, vol. 5, no. 5, pp. 297–309, Sep. 1997.

[5] W. Feng, "Rate-constrained bandwidth smoothing for the delivery of stored video," in *Proc. SPIE Multimedia Netw. Computing*, Feb. 1997, pp. 58–66.

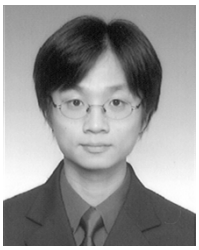
[6] ——, "Time constrained bandwidth smoothing for interactive video-on-demand systems," in *Proc. Int. Conf. Comput. Commun.*, Nov. 1997, pp. 291–302.

[7] R. I. Chang, M. C. Chen, J. M. Ho, and M. T. Ko, "Designing the ON-OFF CBR transmission schedule for jitter-free VBR media playback in real-time networks," in *Proc. 4th Int. Workshop Real-Time Computing Syst. Applicat.*, Oct. 1997, pp. 2–9.

[8] J. D. Salehi, S. L. Zhang, J. Kurose, and D. Towsley, "Supporting stored video: Reducing rate variability and end-to-end resource requirements through optimal smoothing," *IEEE/ACM Trans. Netw.*, vol. 6, no. 4, pp. 397–410, Aug. 1998.

[9] J. Zhang, "Using multiple buffers for smooth VBR video transmissions over the network," in *Proc. Int. Conf. Commun. Technol.*, vol. 1, Oct. 1998, pp. 419–423.

[10] W. Feng and J. Rexford, "Performance evaluation of smoothing algorithms for transmitting prerecorded variable-bit-rate video," *IEEE Trans. Multimedia*, vol. 1, no. 9, pp. 302–312, Sep. 1999.
[11] W. Zhao and S. K. Tripathi, "Bandwidth-efficient continuous media streaming through optimal multiplexing," in *Proc. Int. Conf. Meas. Modeling Comput. Syst.*, Apr. 1999, pp. 13–22.
[12] H. Liu, N. Ansari, and Y. Q. Shi, "Dynamic bandwidth allocation for VBR video traffic based on scene change identification," in *Proc. Int. Conf. Inform. Technol.: Coding Computing*, Mar. 2000, pp. 284–288.
[13] J. Rexford, S. Sen, J. Dey, W. Feng, J. Kurose, J. Stankovic, and D. Towsley, "Online smoothing of live, variable-bit-rate video," in *Proc. Int. Workshop Network Operating Syst. Support Digital Audio Video*, May 1997, pp. 249–258.
[14] S. C. Liew and D. C.-Y. Tse, "A control-theoretic approach to adapting VBR compressed video for transport over a CBR communications channel," *IEEE/ACM Trans. Netw.*, vol. 6, no. 1, pp. 42–55, Feb. 1998.
[15] N. G. Duffield, K. K. Ramakrishnan, and A. R. Reibman, "SAVE: An algorithm for smoothed adaptive video over explicit rate network," *IEEE/ACM Trans. Netw.*, vol. 6, no. 6, pp. 717–728, Dec. 1998.
[16] A. Speetzen *et al.*. Communication Networks Class Library. [Online]. Available: http://www.comnets.rwth-aachen.de/doc/cncl.html
[17] G. Zipf, *Human Behavior and the Principle of Least Effort*. Reading, MA: Addison-Wesley, 1994.



**Hin-lun Lai** received the B.Eng. and M.Phil. degrees in information engineering from the Chinese University of Hong Kong in 1999 and 2001, respectively.

He worked in the same department as a Research Assistant from 2001 to 2002. His research focuses on broad-band multimedia communications and, in particular, video-on-demand systems.

**Jack Yiu-bun Lee** (M'95–SM'03) received the B.Eng. and Ph.D. degrees in information engineering from the Chinese University of Hong Kong in 1993 and 1997, respectively.

He participated in the research and development of commercial video streaming systems from 1997 to 1998 and later joined the Department of Computer Science, Hong Kong University of Science and Technology, as an Assistant Professor from 1998 to 1999. In 1999, he joined the Department of Information Engineering, Chinese University of Hong Kong, where he established the Multimedia Communications Laboratory to conduct research in distributed multimedia systems, fault-tolerant systems, peer-to-peer systems, multicast communications, and Internet computing.

**Lian-kuan Chen** (S'85–M'86–SM'00) received the B.S. degree from National Taiwan University, Taipei, in 1983 and the M.S. and Ph.D. degrees from Columbia University, New York, in 1987 and 1992, respectively, all in electrical engineering.

He was with Jerrold Communication, General Instruments (GI), from 1990 to 1991, where he was engaged in research on linear lightwave video distribution systems, with contributions on distortion reduction schemes for various optical components. He joined the Department of Information Engineering, Chinese University of Hong Kong, in 1992 and initiated the Lightwave Communications Laboratory. He has been the Head of Graduate Division in Information Engineering since 2001. His current interests are in technologies and applications for broad-band local access, photonic signal processing, and the performance management of optical networks. He has published more than 100 papers in international conferences and journals.