Practical Real-Time Approaches for Scheduling VBR Video in a Client-Server Computing Environment

Kelvin Yiu-Lun Tsoi and Yu-Kwong Kwok

Department of Electrical and Electronic Engineering
The University of Hong Kong, Pokfulam Road, Hong Kong

Email: {*yltsoi, ykwok*}*@eee.hku.hk*

Abstract—Scheduling of multiple variable-bit-rate (VBR) video packet streams at the server is one of the most crucial issue in the design of a enterprise video-on-demand (VoD) system. Scheduling is important to guarantee the requested QoS levels while maximizing the utilization of the resources. But the scheduling problem is difficult due to the inherent varying resource requirements of the video streams. Although much work on smoothing and scheduling algorithms has been done, these previous techniques, which usually require the knowledge of future frame sizes and are of a high time complexity, are not suitable because in the video application environment envisioned in our study, real-time response from the server is required. In this paper, we propose and evaluate four real-time approaches using real MPEG data. Our results reveal that the proposed real-time techniques introduce only a slight jitter effect on the video display. Furthermore, time-division based algorithms are superior to rate based ones.

Keywords: multimedia networking, client-server systems, QoS, real-time scheduling, smoothing, VBR video, ATM networks.

I. Introduction

Recently, we have witnessed a spectacular growth in the usage of media data, in particular, video data, in various kinds of information systems [5], [15]. Indeed, video-on-demand (VoD) applications are becoming core utilities in many enterprise intranets, which are largely based on local area network (LAN) environments. To support such applications, powerful servers and a high speed network are needed. Advances in architectural design and reduction in prices have enabled symmetric multiprocessors (SMP) to be commonly used in a cluster of workstations (COW) environment [9] (see Figure 1). The symmetry in shared memory access and the hardware enforced cache coherency features are among the most attractive characteristics of an SMP server. These architectural features make an SMP server easy to program and manage. Indeed, it is now very common to use an SMP as a Web server. In view of the powerful capabilities of an SMP, a broad range of applications, including VoD applications, are under development by our research group. In particular, an important undertaking is to develop a real-time VoD application, two major features of which are real-time video frames delivery upon requests and simultaneous display of multiple clips on a single client machine. These two features imply that a client machine does not have a large buffer and the server has to employ an efficient real-time scheduling technique to schedule video frames. Thus, any high complexity packet transmission strategies are not suitable. In this paper, we propose and evaluate four different practical real-time scheduling techniques for such an application environment.

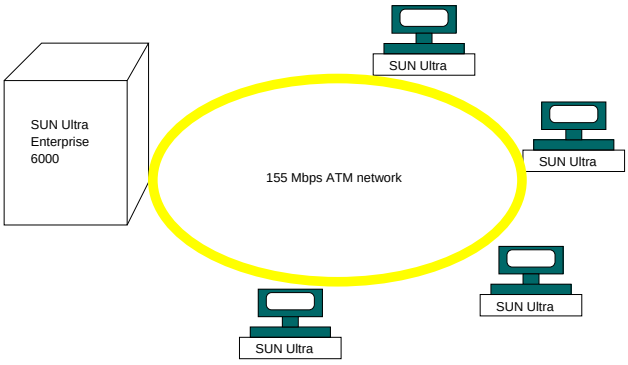


Fig. 1. The client-server environment environment considered in this paper (a cluster of workstations and SMP servers).

One of the most crucial issues in the design of a scheduling algorithm is the support of quality-of-service (QoS) guarantees for real-time transport of stored video over the underlying high speed network

[4], [7], [8], [10]. QoS is usually defined as a tuple of several attributes, such as, delay and packet loss [19]. For smooth continuous playback of video at the client machine, strict compliance of QoS guarantee is needed. On the other hand, from the resource utilization point of view, it is desirable to support as many simultaneous connections as possible. However, these two goals are conflicting. Taking into account the complexity of the scheduling algorithm makes the problem even more complex due to two problem constraints—the video streams are encoded in a constant quality variable-bit-rate (VBR) manner, and the scheduler has only a very limited knowledge about the characteristics of the encoded streams (i.e., the sizes of frames to be sent in the near future). A VBR encoded video stream is much more difficult to be handled than a constant-bit-rate (CBR) one from a resource allocation point of view but a CBR video stream is of a varying display quality which is not desirable. MPEG-2 encoded video streams are the most popular examples of VBR video data. Allocating a fixed rate to such a stream is likely to cause low utilization and/or high packet loss. Many smoothing techniques have been proposed to tackle this problem by rearranging data from one frame to another using buffers [6], [11], [12], [13], [16], [17], [20]. However, such techniques inevitably require knowledge about the sizes of many frames ahead of transmission (e.g., one or more group-of-pictures (GOP) in MPEG-2). This is not possible in the application environment considered in our study due to the real-time retrieval and transmission requirement.

We propose four different real-time scheduling techniques for the transport of VBR video data in an SMP server. The first two algorithms, which employ rate based approaches, work by dynamically adjusting the offered transmission rate of the virtual circuits (VC). The two algorithms differ in the way of adjusting the rates of the simultaneous connections. The other two algorithms are designed based on the time-division concept. Instead of partitioning the total bandwidth into several logical VCs, these two algorithms allow a video request to use the entire bandwidth (e.g., 155 Mbps) to transmit its frames but for a limited amount of time. The next time slot will be used by another request and so on. The sequencing decisions are derived by our enhanced earliest-deadline-first (EDF) and rate-monotonic (RM) real-time scheduling. Our theoretical analysis and experiments using real MPEG encoded data reveal that, contrary to our expectation, time-divison approaches outperform rate based approaches in the small-scale cluster environment we used (an SMP server connected to twelve workstations).

This paper is organized as follows. The next section provides the descriptions of the four proposed techniques. We present the analytical performance of the proposed approaches in Section III. Section IV contains the results of our experiments. The last section concludes the paper.

II. Real-Time Approaches for Scheduling VBR Video

In this section, we first briefly review the general principles behind MPEG encoding and introduce some notations. We then describe the four proposed approaches.

A. MPEG Video Encoding Scheme

In an MPEG encode video stream, there are three types of frames: intraframe coded frames (I frames), predictive interframe coded frames (P frames), and bidirectional interframe coded frames (B frames). An I frame is coded independently in that the whole frame undergoes 8×8 discrete cosine transform (DCT) without referring to any other frames. The DCT coefficients are then quantized and variable-length coded. A P frame is coded based on a preceding I frame using motion compensation techniques in order to reduce the temporal redundancy. A B frame is coded in a similar manner as a P frame but the motion compensation is bidirectional (i.e., using two I or P frames as references). Consequently, a B frame is much smaller than an I frame (by about an order of magnitude). A P frame is also smaller than an I frame but larger than a B frame. In MPEG encoding, a number of frames, typically 12 or 15, are grouped together to form a repeating pattern—a group of pictures (GOP). The structure of a GOP is fixed for a particular sequence, for example, IBBPBBPBBPBB. A GOP is characterized by two parameters: the number of frames, p, and the number of B frames between two successive I or P frames, q. We denote the average sizes of I, P, and B frames as S_I, S_P, and S_B, respectively. Note that although we do not know the sizes of individual frame before retrieval, it is easy to store the average frame sizes (as well as the GOP structure) by the encoder. We assume the display rate at a client machine is f frames per second. Induced by the sizes of buffers dynamically allocated by the client video application (depending upon the number of simultaneous connections), there is a deadline d_i associated with every frame i.

B. QoS Based Scheduling

In our study, we aim at studying the efficacy of two orthogonal approaches—rate based and time-division based—for video packet scheduling in a scalable video

data server in a LAN environment. As in most video data transmission systems, the loss of certain amount of data packets will only lead to degradation of the display quality in the form of jitter (e.g., a frame arrives late) or abrupt motion (e.g., some frames are lost) rather than a catastrophe. Thus, such a system is commonly referred to as a soft real-time environment. Specifically, we assume that during the connection setup phase of a particular video request from a client machine, the client informs the server the level of tolerance of such degradation. In response to the client's request, the server executes an admission control policy (i.e., a schedulability test) in order to determine whether or not the client's request can be accommodated without degrading the current commitments to the existing services. If schedulability test is positive, then the server will start to schedule the requested video; otherwise, the result is sent back to the client and re-negotiation may need to be done. We use the following definition of quality of service (QoS) to quantify the client's requirement of the scheduling of a particular packet stream.

Definition 1: QoS is defined as the probability that an arbitrary video frame meets its display deadline.

Thus, QoS can be interpreted as the expected percentage of frames that meet the transmission deadlines (induced by the display deadlines) at the server. To meet the conflicting goals of high network utilization and strict QoS guarantee, we employ a *lossy* approach in our four proposed algorithms. Specifically, to maintain the display quality while ensuring meeting delay requirement (to prevent buffer underflow), we selectively discard frames that are relatively less important—the B frames. Certainly, if some B frames are missing, some motions in the video sequence will be lost. However, as B frames are not discarded successively but rather sparingly, we do not expect a serious jitter effect in the display. We do not discard data at the ATM cell level because the "blocky" effect [13] is much more undesirable. Furthermore, at the cell level, it is very difficult to differentiate an I frame cell from and B frame cell.

The architecture of video presentation system at a client machine is depicted in Figure 2. We assume that the client has a display buffer of b bits. Thus, after the buffer is full (e.g., before any frame is displayed after the initial connection setup), the maximum *slack* time δ, during which the server does not have to transmit any frame to the client, is given by:

$$\delta = \frac{b}{Sf}$$

where S is the decoded frame size and f is the display rate. However, if the server fails to send any frame to the client after this slack time is elapsed, the client may experience some jitter effect on the screen. The amount of jitter depends on the lateness of the frames. In the following, we refer this lateness to as the *delay* of the video frame.

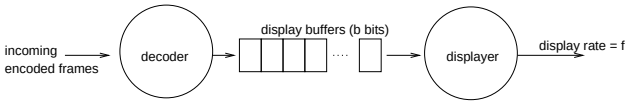


Fig. 2. The architecture of the video presentation system at a client machine.

There are two orthogonal methods for multiplexing a number of simultaneous transmissions onto the outgoing trunk of the SMP server to the ATM network: rate based approach and time-divison based approach. In the former, each VC is assigned a distinct rate which may be changed dynamically. Multiple connections are being served concurrently in this case. In the latter, there is no VC but each connection is assigned a variable length time slot in which the whole bandwidth can be used by the video stream. That is essentially a time-division multiplexing approach. The conceptual difference of the two approaches is illustrated in Figure 3.

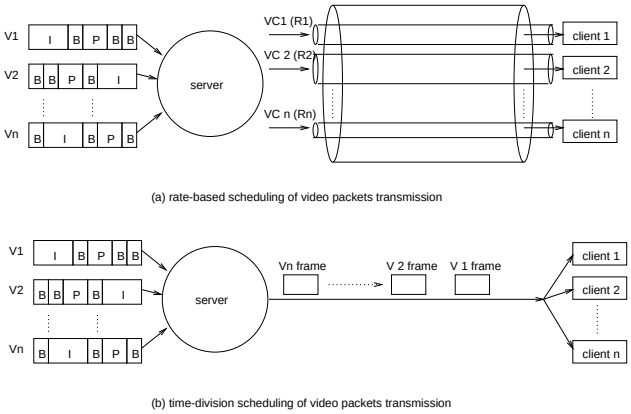


Fig. 3. Two orthogonal schemes for scheduling video frames.

C. Rate Based Approaches

In rate based approaches, the rates allocated to the VCs are dynamically adjusted. The initial rate of a VC is determined as follows. Consider a long sequence of L frames. Thus, the number of GOPs in the sequence is L/p. Hence, the number of I frames to transmit is also L/p. The number of P frames to transmit is

$$\frac{L}{p}(\frac{p}{q} - 1).$$

The number of B frames to transmit is

$$(q-1)\frac{p}{q}\frac{L}{p}.$$

The total number of bits to transmit is

$$S_I\frac{L}{p} + S_P(\frac{p}{q}-1)\frac{L}{p} + S_B L(1-\frac{1}{q}).$$

Thus, the expected average rate (initial rate) is given by:

$$R_0 = f\left(\frac{S_I}{p} + \frac{S_P}{p}(\frac{p}{q}-1) + S_B(1-\frac{1}{q})\right).$$

The two proposed rate based approaches employ radically different strategies to adjust the rates. The first algorithm is based on a demand oriented approach such that the rate of a stream, which has more deadline missed, is decreased. The rationale is that the stream transmitting significantly larger frames than others is penalized. In the second algorithm, we use a supply oriented approach in which the rate of a stream with more deadline misses is increased. This is done in order to provide higher bandwidth to satisfy the temporary increase in frame sizes. Our first proposed rate based scheduling algorithm, called RBS1, is outlined below.

RBS1:

(1) FOR each video stream i:
    set initial rate $R^i{}_0$;
(2) DO WHILE there are frames to transmit:
(3) FOR each video stream i:
(4) if current rate cannot support transmitting the current frame:
(5) if it is an I or P frame, transmit it (deadline may be missed);
(6) if it is an B frame, discard it;
(7) record the number of frames m_i in each stream i that miss the deadlines;
(8) in descending order of m_i, proportionately decrease the rates;
(note that a stream with a smaller value of m may get an increased rate)

The RBS1 algorithm only needs a linear time to decide the new rate (the last step) and, thus, can be executed in real-time for a moderate number of requests (e.g., less than 25 connections as considered in our study). The second rate based algorithm RBS2 differs from RBS1 only in the last step in which the rate adjustment is done in ascending order of number of deadline misses. Note that in both approaches, admission control is easy to implement: just check whether the total bit rate exceeds the bandwidth of the network.

D. Time-Division Based Approaches

Our next two proposed approaches are based on the time-division concept. In these two schemes, we treat each video stream as a periodic request such that the period T is just $1/f$. This is depicted in Figure 4. However, as mentioned earlier, if a frame of a certain connection is scheduled to transmit, it can use the whole bandwidth for the transmission. Thus, the rate is constant but the service time is varying (a larger frame needs longer time to transmit). For deciding which particular stream to go first, we employ our two previously developed real-time scheduling techniques for scheduling CBR traffics, called QEDF and QRM [18]. These two algorithms are enhanced versions of the classical EDF and RM algorithms [14]. In the QEDF algorithm, the frame among all streams that has the earliest deadline is scheduled to be transmitted first. However, if such a frame is large (e.g., an I frame), it may cause deadline misses in subsequent waiting frames. Among all these potential deadline missing frames, B frames are discarded. In the QRM algorithm, the stream with the highest display rate is selected to go first. Again, for potential deadline missing frames, B frames are discarded. Since both QEDF and QRM are relatively straightforward, they can also be executed in real-time.

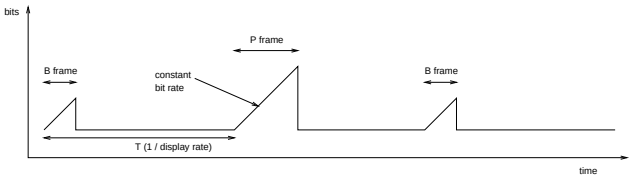


Fig. 4. Scheduling video frames using a periodic approach.

In QEDF and QRM scheduling, the scheduling decisions are based on the display rates of the video streams. In the following, we assume that there are n video streams with display rates $f_1 \geq f_2 \geq \cdots \geq f_n$ so that the periods are $T_1 \leq T_2 \leq \cdots \leq T_n$. For EDF scheduling, consider an arbitrary frame F_{ij} of the i-th video stream. Note that the ready time of F_{ij} is $(j-1)T_i$ and deadline is jT_i. The number of frames whose deadlines are before jT_i is given by:

$$\sum_{\forall k, k\neq i} \left\lfloor \frac{jT_i}{T_k} \right\rfloor$$

and the number of frames transmitted before $(j-1)T_i$ is given by:

$$\sum_{\forall k, k\neq i} \left\lfloor \frac{(j-1)T_i}{T_k} \right\rfloor$$

Thus, during the time period from $(j-1)T_i$ to jT_i, the number of frames need to be transmitted:

$$\sum_{\forall k, k\neq i} \left(\left\lfloor \frac{jT_i}{T_k} \right\rfloor - \left\lfloor \frac{(j-1)T_i}{T_k} \right\rfloor \right)$$

Using the relation $x \geq \lfloor x \rfloor \geq x-1$, the maximum number of frames that can be transmitted is:

$$\sum_{\forall k, k \neq i} \left(1 + \frac{T_i}{T_k}\right) \quad (1)$$

For RM scheduling, using similar arguments as above, the number of frames to be transmitted between $(j-1)T_i$ and jT_i is:

$$\sum_{k<i} \left(\left\lceil \frac{jT_i}{T_k} \right\rceil - \left\lfloor \frac{(j-1)T_i}{T_k} \right\rfloor \right)$$

Again, using the relations $x \geq \lfloor x \rfloor \geq x-1$ and $x+1 \geq \lceil x \rceil \geq x$, the maximum number of frames that can be transmitted is:

$$\sum_{k<i} \left(2 + \frac{T_i}{T_k}\right) \quad (2)$$

The admission control procedure of time-division based approaches is more complex than that of the rate based approaches. In the following admission control algorithm, which checks whether a new video request x can be admitted, we assume in the worst case m frames are to be transmitted within one transmission period (m is given by (1) and (2) for QEDF and QRM, respectively). We use R to denote the total bandwidth.

Admission Control:
t_I, t_P, t_B: time required to transmit I, P, and B frames of existing video streams
(1) $t_I = 0$, $t_P = 0$, $t_B = 0$
(2) FOR every existing video stream k
(3) $\quad t_I = t_I + S_I^k/R$
(4) $\quad t_P = t_P + (\lceil m/q \rceil - 1)S_P^k/R$
(5) $\quad t_B = t_B + (m - \lceil m/q \rceil)S_B^k/R$
(6) $\tau = t_I + t_P + t_B$
(7) if $(T_x - \tau - S_I^x/R + \delta > 0)$ then video x can be admitted

III. Analysis

In this section, we analyze the performance of the two orthogonal approaches. In the transmission of video streams, the average slack time is a crucial performance parameter because it indicates how much extra capacity the scheduler is able to allow for new video requests. We prove that the time-division based approach is in general better than the rate based approach.

Consider frame F_{ij}, the j-th frame of the i-th video stream. The following notations are used in the analysis:

- S_{ij}: size of F_{ij}
- e_{ij}: finish time of transmitting F_{ij}
- r_{ij}: ready time of F_{ij}, that is, $\max(e_{i-1,j}, (j-1)/f)$
- d_{ij}: deadline of F_{ij}
- R_i: transmission rate of video i
- $\Delta_{ij} = d_{ij} - e_{ij}$ (i.e., the transmission slack time)

Definition 2: Given two packet scheduling algorithms $\mathcal{A}$ and $\mathcal{A}'$, we say that $\mathcal{A}$ outperform $\mathcal{A}'$ if $\mathcal{A}$ gives a larger value of $\sum \Delta_{ij}$.

Theorem 1: Time-division based algorithms (QEDF and QRM) outperform rate based algorithms (RBS1 and RBS2).

Proof: We prove the theorem by using mathematical induction. First consider two video streams. Let F_1 and F_2 be two arbitrary frames from the two streams, respectively. The first subscript is dropped for clarity. Figure 5 depicts the two scheduling strategies. For rate based approach, we have:

$$\begin{aligned} \Delta_1 + \Delta_2 &= d_1 - e_1 + d_2 - e_2 \\ &= d_1 + d_2 - \frac{S_1}{R_1} - \frac{S_2}{R_2} \end{aligned} \quad (3)$$

For time-division based approach, we have:

$$\Delta_1' + \Delta_2' = d_1 + d_2 - \frac{S_1}{R_1 + R_2} - \frac{S_1 + S_2}{R_1 + R_2} \quad (4)$$

Without loss of generality, assume:

$$\frac{S_1}{R_1} \leq \frac{S_2}{R_2}$$

Then, we have:

$$\frac{S_1 + S_2}{R_1 + R_2} = \left(\frac{S_2}{R_2}\right)\left(\frac{S_1/S_2 + 1}{R_1/R_2 + 1}\right) \leq \frac{S_2}{R_2}$$

Thus, we have:

$$\Delta_1' + \Delta_2' > \Delta_1 + \Delta_2$$

The basis case is true and let us make the following induction assumption (for $k \geq 2$):

$$\begin{aligned} d_1 + \cdots + d_k - \tfrac{S_1}{R_1 + \cdots + R_k} - \tfrac{S_1 + \cdots + S_k}{R_1 + \cdots + R_k} \geq \\ d_1 + \cdots + d_k - \tfrac{S_1}{R_1} - \cdots - \tfrac{S_k}{R_k} \end{aligned} \quad (5)$$

Consider the case where there are $k+1$ video streams. We need to prove:

$$\begin{aligned} \tfrac{S_1}{R_1 + \cdots + R_{k+1}} + \cdots + \tfrac{S_1 + \cdots + S_{k+1}}{R_1 + \cdots + R_{k+1}} \leq \\ \tfrac{S_1}{R_1} + \cdots + \tfrac{S_{k+1}}{R_{k+1}} \end{aligned} \quad (6)$$

Now, L.H.S. of (6):

$$\begin{aligned} \tfrac{S_1}{R_1 + \cdots + R_{k+1}} + \cdots + \tfrac{S_1 + \cdots + S_{k+1}}{R_1 + \cdots + R_{k+1}} \leq \\ \tfrac{S_1}{R_1 + \cdots + R_k} + \cdots + \tfrac{S_1 + \cdots + S_k}{R_1 + \cdots + R_k} + \tfrac{S_1 + \cdots + S_{k+1}}{R_1 + \cdots + R_{k+1}} \end{aligned} \quad (7)$$

By (5), the above expression is smaller than:

$$\frac{S_1}{R_1} + \cdots + \frac{S_k}{R_k} + \frac{S_1 + \cdots + S_{k+1}}{R_1 + \cdots + R_{k+1}}$$

Thus, it remains to prove:

$$\frac{S_1 + \cdots + S_{k+1}}{R_1 + \cdots + R_{k+1}} \leq \frac{S_{k+1}}{R_{k+1}} \tag{8}$$

Without loss of generality, we assume:

$$\frac{S_1}{R_1} \leq \frac{S_2}{R_2} \leq \cdots \leq \frac{S_{k+1}}{R_{k+1}}$$

Thus, we have:

$$\begin{aligned} \frac{S_1 + \cdots + S_{k+1}}{R_1 + \cdots + R_{k+1}} &= (\frac{S_{k+1}}{R_{k+1}})(\frac{S_1/S_{k+1} + \cdots + 1}{R_1/R_{k+1} + \cdots + 1}) \\ &\leq \frac{S_{k+1}}{R_{k+1}} \end{aligned} \tag{9}$$

The theorem is proved. (Q.E.D.)

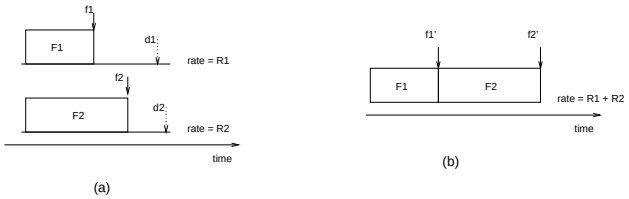


Fig. 5. (a) Rate based scheduling; (b) time-division based scheduling.

IV. Experimental Results

In this section, we present the preliminary results of testing our implementations of the four algorithms in our cluster environment. We used video sequences in CCIR-601 format (720 x 486 pixels) which is about 1 MB per frame [1], [2]. The average peak rate of the encoded sequences is about 15 Mbps. In our first experiment, we tested the transmission of two video streams on a 10 Mbps channel. The characteristics of the two streams, called V_1 (a football clip) and V_2 (a garden clip) are as follows:

- $p = 15$ and $q = 3$
- number of frames: 1000
- average frame sizes of V_1: I frame is 450 Kb, P frame is 290 Kb, and B frame is 130 Kb
- average frame sizes of V_2: I frame is 600 Kb, P frame is 298 Kb, and B frame is 108 Kb
- QoS requested is 1.0 for both streams

Table I shows the average QoS achieved by using the four approaches. The results concur with our analysis in Section III in that the time-division based approaches outperform the rate based approaches.

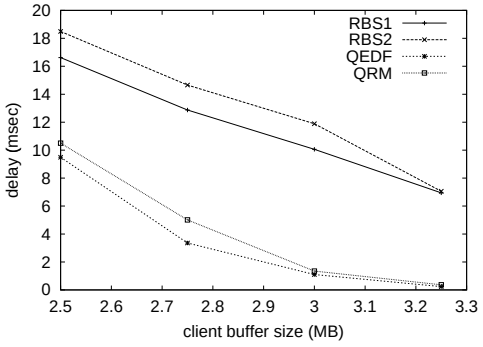
In order to investigate the practicality of the four approaches in a real client-server environment, we used a larger number of video streams. Since our ATM network has a bandwidth of only 155 Mbps, 12 simultaneous requests from various client machines can slightly overload the network if no discarding is used. Thus, to evaluate the performance of the four approaches, we performed experiments using 12, 15, and 20 streams. Obviously, client machine buffer sizes are critical parameters. Based on our video application environment, we used 2.5 MB, 2.75 MB, 3 MB, and 3.25 MB as the buffer sizes. Again, the QoS requested by each stream is 1.

TABLE I
Average QoS results of transmitting for the two video streams.

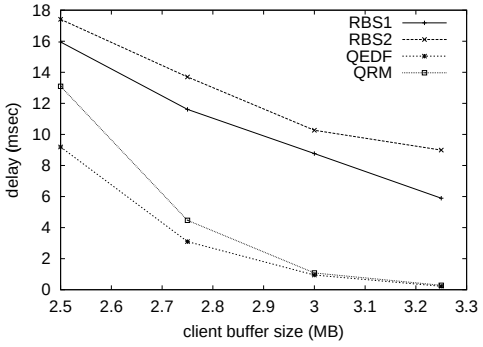
buffer size (MB)	RBS1	RBS2	QEDF	QRM
2.50	0.86	0.84	0.93	0.94
2.75	0.88	0.86	0.95	0.96
3.00	0.90	0.87	0.96	0.96
3.25	0.92	0.88	0.97	0.97

Figure 6 shows the results on average delay, which is defined as the average amount of time a client machine experiences a buffer underflow (see Section II-B). As expected, the delay decreases with the client buffer sizes. The time-division approaches are clearly superior to the rate based approaches. A plausible reason is that in the rate based approaches, because the rate is fixed and limited when a new frame is retrieved, more delay is introduced due to the transmission at a lower than enough rate. This will never happen in a time-division based approach because the full bandwidth can be utilized. From the results, we observe that all four approaches introduce acceptable level of jitter effect (caused by buffer underflow) because the delay is always smaller than the frame period (about 40 msec). Furthermore, with increasing number of streams, the delay only increases slowly. This implies that all the four approaches are quite robust.

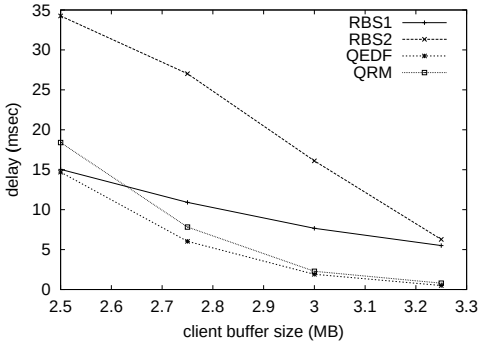
In our next experiment, we used 20 video streams and fixed the buffer size as 2.5 MB. To investigate the QoS performance of the scheduling algorithms, we varied the QoS requested by the clients. We tested four scenarios in which the mean QoS of the 20 video streams were 0.6, 0.7, 0.8, and 0.9. We measured the actual QoS achieved for each video stream and computed the average for each scenario. These results are shown in Figure 7. As can be seen, the rate based approaches inherently do not take into account the client's requested QoS when making rates adjustments. The requested QoS just serves as a lower bound in determining the transmission rate. Thus, the two rate based approaches gave rather constant QoS. On the other hand, the two time-division based approaches are more robust to the requested traffic



(a) 12 video clips.



(b) 15 video clips.



(c) 20 video clips.

Fig. 6. Delay performance of the four different real-time video packet scheduling approaches with different client buffer sizes.

load. This also explains why the time-division based approaches can give smaller delays and hence much less jitter effects. Indeed, when viewed with our decoding and presentation module, the delay and even loss of frames were barely noticeable, even for video with vigorous motion as in the football clip shown in Figure 8.

V. Conclusions

In this paper, we consider the video packet scheduling problem in a cluster of workstations environment. As real-time response is needed from the server and the buffer size of a client machine is limited, efficient lossy techniques are called for. We propose four algorithms, two of which designed based on rate based approach, while the other two are enhanced classical real-time scheduling algorithms. In our theoretical analysis and experimental evaluation, we find that the four algorithms introduce only a slight jitter effect on the video display. Furthermore, time-division based approaches give better loss and delay performance over rate based approaches. However, the implementation of time-division based approaches is more difficult. Another avenue of further research is to investigate the effectiveness of the enhanced scheduling schemes in a wide-area distributed VoD system such as the one considered in [3].

Acknowledgments—The authors would like to thank the insightful comments of the referees. This research was support by a grant from the HKU CRCG. Kelvin Tsoi was also supported by a studentship from the EEE Department at HKU.

References

[1] S.M. Akramullah, I. Ahmad, M.L. Liou, "A Data-Parallel Approach for Real-Time MPEG-2 Video Encoding," *Journal of Parallel and Distributed Computing*, vol. 30, no. 2, pp. 129–146, Nov. 1995.

[2] S.M. Akramullah, I. Ahmad, M.L. Liou, "Performance of Software-Based MPEG-2 Video Encoder on Parallel and Distributed Systems," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, no. 4, pp. 687–695, Aug. 1997.

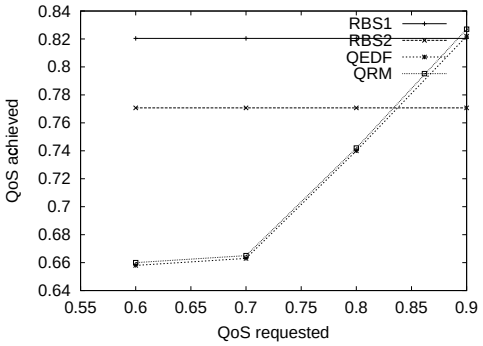
[3] S. Bakiras and V.O.K. Li, "Smoothing and Prefetching for VBR Video-on-Demand Systems with Distributed Video Servers," submitted to *SIGCOMM'99*.

[4] C. Blondia and O. Casals, "Performance Analysis of Statistical Multiplexing of VBR Sources," *Proc. INFOCOM'92*, pp. 828–838.

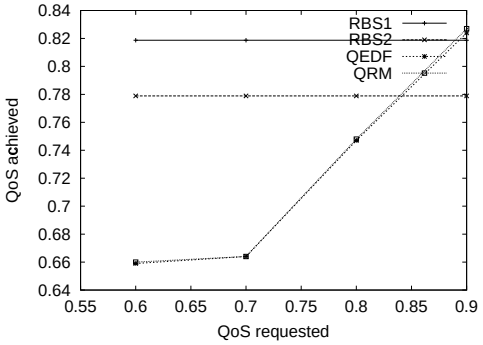
[5] İ. Dalgiç and F.A. Tobagi, "Performance Evaluation of ATM Networks Carrying Constant and Variable Bit-Rate Video Traffic," *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 6, pp. 1115–1131, Aug. 1997.

[6] N.G. Duffield, K.K. Ramakrishnan, and A.R. Reibman, "SAVE: An Algorithm for Smoothed Adaptive Video Over Explicit Rate Networks," *IEEE/ACM Transactions on Networking*, vol. 6, no. 6, pp. 717–728, Dec. 1998.

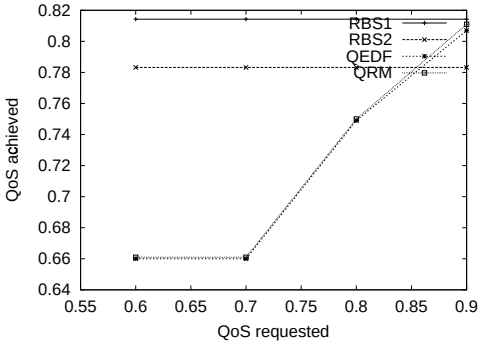
[7] V. Firoiu, J. Kurose, and D. Towsley, "Efficient Admission Control of Piecewise Linear Traffic Envelops at EDF Schedulers," *IEEE/ACM Transactions on Networking*, vol. 6, no. 5, pp. 558–570, Oct. 1998.



(a) 12 video clips.



(b) 15 video clips.



(c) 20 video clips.

Fig. 7. QoS performance of the four different real-time video packet scheduling approaches with different client buffer sizes.

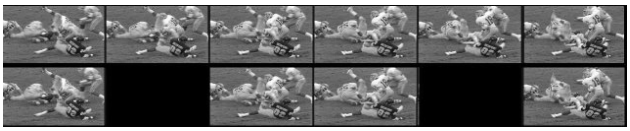


Fig. 8. Illustration of deleting B frames in a video sequence (upper: original; lower: two B frames deleted).

[8] L. Georgiadis, R. Guerin, and A. Parekh, "Optimal Multiplexing on a Single Link: Delay and Buffer Requirements," *IEEE Transactions on Information Theory*, vol. 43, no. 5, pp. 1518–1535, Sept. 1997.

[9] K. Hwang, H. Jin, E. Chow, C.L. Wang, and Z. Xu, "Designing SSI Clusters with Hierachical Checkpointing and Single I/O Space," *IEEE Concurrency*, vol. 7, no. 1, pp. 60–69, Jan.–Mar. 1999.

[10] J.M. Hyman, A.A. Lazar, and G. Pacifici, "Real-Time Scheduling with Quality of Service Constraints," *IEEE Journal on Selected Areas in Communications*, vol. 9, no. 7, pp. 1052–1063, Sept. 1991.

[11] S.S. Lam, S. Chow, and D.K.Y. Yau, "A Lossless Smoothing Algorithm for Compressed Video," *IEEE/ACM Transactions on Networking*, vol. 4, no. 5, pp. 697–708, Oct. 1996.

[12] S.C. Liew and H.H. Chan, "Lossless Aggregation: A Scheme for Transmitting Multiple Stored VBR Video Streams over a Shared Communications Channel Without Loss of Image Quality," *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 6, pp. 1181–1189, Aug. 1997.

[13] S.C. Liew and C.-Y. Tse, "Video Aggregation: Adapting Video Traffic for Transport Over Broadband Networks by Integrating Data Compression and Statistical Multiplexing," *IEEE Journal on Selected Areas in Communications*, vol. 14, no. 6, pp. 1123–1137, Aug. 1996.

[14] C.L. Liu and J.W. Layland, "Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment," *Journal of the ACM*, vol. 20, no. 1, pp. 46–61, Jan. 1973.

[15] J.M. McManus and K.W. Ross, "Video-on-Demand Over ATM: Constant-Rate Transmission and Transport," *IEEE Journal on Selected Areas in Communications*, vol. 14, no. 6, pp. 1087–1098, Aug. 1996.

[16] D. Reininger, D. Raychaudhuri, B. Melamed, B. Sengupta, and J. Hill, "Statistical Multiplexing of VBR MPEG Compressed Video on ATM Networks," *Proc. INFOCOM'93*, pp. 919–926.

[17] J.D. Salehi, Z.-L. Zhang, J. Kurose, and D. Towsley, "Supporting Stored Video: Reducing Rate Variability and End-to-End Resource Requirements Through Optimal Smoothing," *IEEE/ACM Transactions on Networking*, vol. 6, no. 4, pp. 397–410, Aug. 1998.

[18] K.Y.-L. Tsoi and Y.-K. Kwok, "Meta-QoS Performance of Earliest-Deadline-First and Rate-Monotonic Scheduling of Smoothed Video Data in a Client-Server Environment," *Proceedings of the 1999 International Symposium on Parallel Architectures, Algorithms, and Networks (I-SPAN'99)*, Fremantle, Western Australia, June 1999, accepted for publication and to appear.

[19] D.E. Wrege, E.W. Knightly, H. Zhang, and J. Liebeherr, "Deterministic Delay Bounds for VBR Video in Packet-Switching Networks: Fundamental Limits and Practical Trade-Offs," *IEEE/ACM Transactions on Networking*, vol. 4, no. 3, pp. 352–362, June 1996.

[20] Z.-L. Zhang, J. Kurose, J.D. Salehi, and D. Towsley, "Smoothing, Statistical Multiplexing, and Call Admission Control for Stored Video," *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 6, pp. 1148–1166, Aug. 1997.