Traffic-Smoothing for Delivery of Online VBR Media Streams by a Dynamic Window-Based Approach

Ray-I Chang

Meng-Chang Chen

Jan-Ming Ho

Ming-Tat Ko

Institute of Information Science

Academia Sinica

Taipei, Taiwan, ROC

{william, mcc, hoho, mtko}@iis.sinica.edu.tw

Abstract

Traffic smoothing for delivery of online VBR media streams is one of the most important problems in designing multimedia systems. Given the available client buffer and the sliding smooth window, conventional approaches try to minimize bandwidth allocated in each window. However, they can not lead to the minimization of bandwidth allocated for transmitting the entire stream. Although a sliding-window approach is introduced by Rexford et al. in 1997 to further reduce the bandwidth allocated, it is time-consuming. To resolve these drawbacks, in this paper, an effective and efficient online traffic-smoothing scheme is proposed. Different from conventional constant-sized approaches, our approach can automatically decide the suitable sliding sizes to online smooth the burst VBR traffic. By examining various media streams, our approach is shown to have higher bandwidth utilization (or called bandwidth-occupancy in conventional approaches) and requires smaller bandwidth than conventional approaches. Considering the online transmission of a *Star War* movie, our obtained result is over 13% smaller in the required network bandwidth and over 4% smaller in the obtained network idle rate than conventional approaches. In this paper, a feedback control method is introduced to resolve the latency- and quality-tolerance applications. Besides, the relations between the characteristic of input traffic and the behavior of obtained scheduling results are also discussed.

1. Introduction

In a distributed multimedia system, VBR media streams in servers should be smoothed and transmitted across network to clients by following a transmission schedule. To guarantee media QoS (quality-of-service), the obtained transmission schedule should satisfy real-time requirements for jitter-free playback. Besides, the allocated resources (such as network bandwidth) should be minimized and fully utilized to support as many requests as possible. Generally, a network transmission schedule can be categorized as either client-controlled [1] or server-controlled [2-9]. In a client-controlled transmission scheme, the utilization of client buffer can be monitored with better knowledge about the client status. However, the client still need to send feedback messages to servers. As the network is possibly congested, this client-controlled scheme has the drawback of feedback overhead and the responses may be dramatically delayed. In these years, server-controlled transmission schemes have received great attentions for delivery of VBR media streams to guarantee deterministic network services. If the media stream is pre-recorded, the entire traffic can be accurately characterized to allocate appropriate mount of resources and minimize the specified cost functions (*i.e.* critical bandwidth, bandwidth changes or bandwidth variability) [2-15]. The obtained smoothed traffic usually has smaller transmission overhead than the original VBR traffic. Notably, if the VBR traffic were generated online, the characteristics of online traffic would not be precisely analyzed offline as presented in [16, 21]. Although some prediction methods are presented to estimate the sizes of future frames based on their previous frames [19-20], the prediction may not be correct.

In 1997, the window-based traffic-smoothing approach was introduced for delivery of online media streams. Given a suitable delay as the time window size (called *delay window*), media frames in this time window are specified and have not been transmitted. As frame sizes in time windows are given, the smoothing algorithms presented for pre-recorded VBR media streams can be applied to smooth the traffic in this time window. In [21], the CBA (critical bandwidth allocation) method (which proposed in [6] to minimize the allocated bandwidth for a pre-recorded media stream) is directly applied in each time window to smooth online burst traffics. They smooth the traffics between different time windows independently. Note that, in traffic smoothing, the large-size frames are pre-transmitted ahead of their playback time to reduce required bandwidth. Let's consider a burst VBR media stream, in which, a sequence of small-size frames (called window w_1) is followed by an action-scene with large-size frames (called window w_2). This kind of burst traffic can be found in many VBR media streams. If the conventional hopping-window scheme [21] is applied, as w_1 and w_2 are mutually exclusive, the large-size frames in w_2 could not be pre-transmitted to smooth this burst traffic. Although bandwidths allocated in windows w_1 and w_2 are minimized, they do not lead to the minimization of bandwidth required for transmitting the entire stream. Recently, the sliding-window approach with a constant sliding size [21] is introduced to further reduce the peak bandwidth allocated by overlapping the considered time windows. However, it is hard to decide the best window sliding size to smooth the peak bandwidth. Generally, if the sliding size were large, the obtained improvements would be limited. Besides, it would be very time-consuming if the sliding size were too small. Although the increasing of time window size may reduce the required peak bandwidth, the delay time for media playback would be highly increased. This assumption is not acceptable for some real-time applications.

To resolve these drawbacks, in this paper, a dynamic window-sliding scheme is proposed. Our approach can automatically decide the suitable sliding size of time window to smooth VBR traffic. Thus, the large-size frames would be pre-transmitted as early as possible to reduce the required bandwidth. In this paper, we have examined our proposed approach by different real-world streams. Experiments show that our proposed method is effective and efficient. Given the client buffer size and the delay window size, our proposed approach requires smaller network bandwidth and can obtain higher network utilization than the conventional approaches. Our required computation time is also smaller than that of the one-frame sliding-window approach. If network congestion or buffer overflow is detected, we introduce a feedback control algorithm to decrease the encoding speed and the encoding quality to reduce the required bandwidth. Otherwise, based on the transmission deadline computed by [2], the suitable frames can be dropped or deferred for playback to fit the available bandwidth. This bounded bandwidth allocation technique is very useful when considering a real-world application with the latency- and quality-tolerance [10].

2. Delivery of Online VBR Media Streams

At client sites, media frames are played periodically. The related playback schedule can be described by the cumulative sizes of media frames called *cumulative playback function (CPF)* [2-3]. Assume that a media stream is defined as $V = \{ f_0 f_1 \ldots f_{n-1}; T_f \}$ where f_i is the i-th media frame size and T_f is the period time for frame playback. n is the number of frames in this media stream. Assume that the client starts playback at time 0. The cumulative playback function $F(t)$ at time t is defined as follows.

$$
\begin{aligned}
F(t) &= 0, && \text{if } t < 0 \\
&= F_i, && \text{if } i*T_f \le t < (i+1)*T_f, \text{ where } 0 \le i < (n-1) \\
&= F_n, && \text{if } (n-1)*T_f \le t
\end{aligned}
$$

$F_i = F_{i-1} + f_i$ is the i-th cumulative media size (where $0 \le i < n$ and $F_{-1} = 0$). As $F(t)$ is a stair function, we can define $F(i*T_f)^-$ and $F(i*T_f)^+$ as the lower and the upper corner values at time $i*T_f$. Assume that the server starts transmission at time $-d*T_f$, $d*T_f$ is called the *delay time*. Assume that the transmission rate r_i is applied to transmit the media stream between time $(i*T_f)$ to time $((i+1)*T_f)$. A transmission schedule can be identified by a sequence of transmission rates $r_{-d} r_{-d+1} \ldots r_0 r_1 \ldots r_{n-2}$ or a sequence of transmission segments $<p, q, r_p>$ where $r_p = r_{p+1} = \ldots = r_{q-1}$. The peak bandwidth allocated for media transmission is defined as $\max\{ r_i : \text{for all } i \}$.

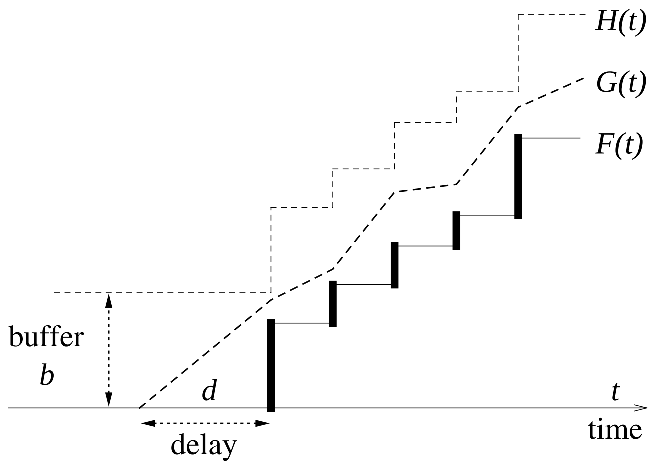


Fig. 1. A jitter-free transmission schedule for the given client buffer size and initial delay. The design goal of a good transmission schedule is to smooth the VBR traffic with the minimum

2.1. Jitter-Free Transmission Schedule

Based on the definition of CPF, we define the *cumulative transmission function* $G(t) = G((i-1)*T_f) + r_i*T_f$ as the amount of data sent by the transmitter. Notably, a jitter-free transmission schedule should ensure a complete data frame is transmitted before each display without buffer overflow and underflow. As the consumed data $F(t)$ is a stair function and the buffer size b is bounded, the cumulative data transmitted before time t should not be larger than $H(t) = \min\{ F_n, F(t) + b \}$. Besides, $G(t)$ should not be smaller than $F(t)$ for continuous playback. The jitter-free transmission schedule $G(t)$ should satisfy $F(t)^+ \leq G(t) \leq H(t)^-$ as shown in Fig. 1. It can be easily proved that, for all i, $f_i \leq b$. For different jitter-free transmission schedules, their performances can be measured by the required resources and the obtained resource utilization. Generally, the resources considered in a multimedia system could be the *memory buffer* ands the *network bandwidth*. As shown in previous studies [2-9], large initial delay and client buffer can act as a good reservoir to regulate the difference between transmission and playback rates. However, in a real-world system, the available buffer size and the acceptable delay time are bounded and highly dependent on the service provided. In [2], we present a linear-time transmission schedule algorithm for delivery of pre-recorded VBR media stream. An O($n$log$n$) algorithm presented to offline compute the relation functions between the required resources is shown in [18]. Based on these relation functions, the session set-up protocol is as simple as a request-reply with a constant time computation. Although this approach has been proved optimal on the required resource and the obtained resource utilization, it is not suitable for delivery of online generated media streams.

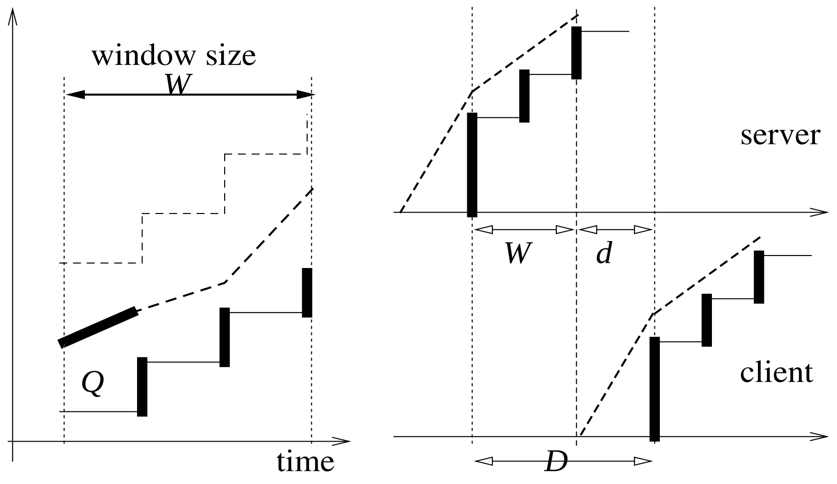


Fig. 2. A simple example to demonstrate the differences between the pre-recorded media stream and the online generated media stream. Notably, in a window-based problem model, the frames outside the current window are already scheduled or not generated.

2.2. Online-Generated VBR Media Stream

In a distributed multimedia system, media data can be pre-recorded or online-generated. The pre-recorded media data are generally stored in disks [24]. Based on the data layout, a disk retrieval scheduler [22-23, 28] is applied to decide the retrieving time of data blocks. Then, a network transmission schedule is designed to minimize the required resource to support as many requests as possible. In a pre-recorded media stream, as all the frame sizes are specified, the transmission schedule can be computed offline [2-9]. However, in an online generated media stream, the frame sizes are not pre-specified before session setup. At any time *t*, the previous frames may be transmitted and the future frames are not generated. The transmission schedule can only be decided by a time window of functions

F(t) and *H(t)* [2-21]. Although some prediction methods are presented to estimate the sizes of upcoming frames by the previous frames [19-20], the prediction may be not correct. Recently, the window-based traffic model is applied to smooth the online media stream by introducing a small playback delay [21]. A simple example to illustrate the differences between the pre-recorded media stream and the window-based online media stream is shown in Fig. 2. In the window-based model with a window size W, the media frames $f_i f_{i+1} \ldots f_{i+W-1}$ are given for scheduling with $W*T_f$ delay time. The initial delay for playback is $D = (d+W)*T_f$ where d is usually given as one. Notably, as frames before f_i are already transmitted and frames after f_{i+W-1} are not generated, only the frames in this window can be applied to smooth the VBR traffic. We can simply treat the media frames in each time window as pre-recorded. Thus, the traffic smoothing methods presented for the pre-recorded streams (such as CBA [6], MVBA [5], LA [2] and CRTT [9]) can be applied in these time windows. In conventional approaches [21], traffic smoothing for different time windows are operated independent. Although the peak bandwidths allocated in each time window are minimized, the peak bandwidth allocated for transmitting the entire media stream is not minimized. A simple example to demonstrate this drawback is given in Fig. 3. Define that the window w contains only small-size frames and the large-size frames are in the next window w^+. If the traffics in w and w^+ are smoothed independently, the required bandwidth in w is minimized without considering the allocated bandwidth in w^- and the burst traffic in w^+. Thus, the peak bandwidth allocated in w^- would not be specified in w to smooth the burst traffic in w^+. Notably, if the bandwidth allocated in w^- is considered in w and w^+, a better schedule can be obtained as shown in Fig. 3. Our proposed algorithm is based on this concept to improve the required peak bandwidth and the obtained network idle rate.

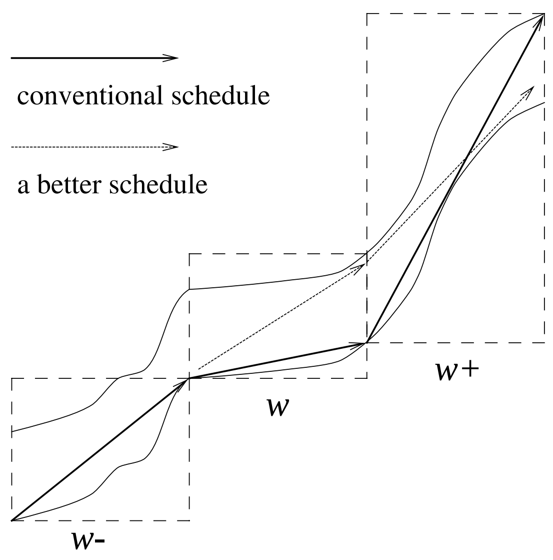


Fig. 3. A simple example to show the drawbacks of conventional window-based approaches. Notably, although the bandwidth allocated in each local window is minimized, the peak bandwidth allocated for transmitting the entire media stream is not minimized.

In [21], a sliding window-based approach is proposed to reshape the media traffic whenever α new frames are generated (called SLWIN(α) where α is the number of sliding frames). Given an acceptable delay time D to decide a suitable window size W, SLWIN(α) first considers the window w_0 with frames $f_0 f_1$ ... f_{W-1}. At time W^*T_f, the scheduling algorithm is executed and starts the transmission schedule. The client receives frame f_0 after d^*T_f pre-loading time and starts the playback. At time $(W+\alpha)^*T_f$, the frames $f_W f_{W+1}$ ... $f_{W+\alpha-1}$ are generated. The scheduling algorithm is executed again to shape the remainder traffic of $f_0 f_1$ ... f_{W-1} and these α new added frames. This α-frame sliding-window approach should run the traffic smoothing algorithm n/α times where n is the number of media frames. It is very time-consuming. Sometimes, the over-determined bandwidth reduction may defeat the pre-transmission of media frames. Although the required bandwidth in the current window is reduced, the peak bandwidth allocated for the entire media stream is increased and the obtained resource utilization is low. It is necessary to design a new algorithm to further minimize the peak bandwidth allocated for delivery of online media streams with low time complexity and high resource utilization.

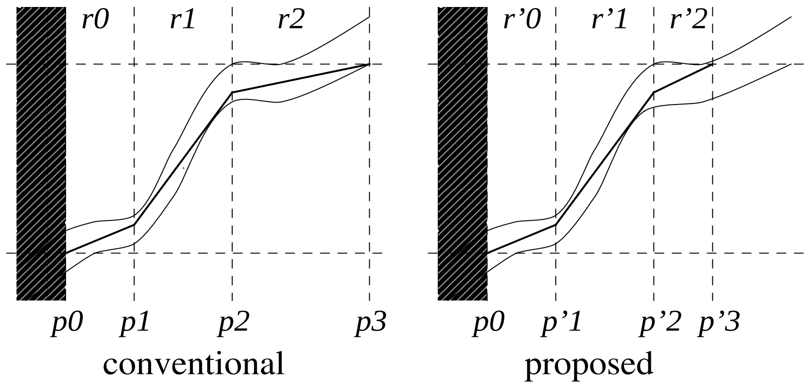


Fig. 4. Compares the peak bandwidth allocated by the conventional approaches and our proposed approach. In our proposed approach, we can fully utilize the peak bandwidth allocated

3. Our Proposed Dynamic Window Sliding Approach

In this paper, an online traffic-smoothing method is proposed to automatically decide the suitable size for window sliding. Different from the conventional approaches, our bandwidth minimization procedures for different windows are operated dependently. In each window, the peak bandwidth allocated in the previous window (called r_{max}) is considered as an input threshold parameter. The initial value of r_{max} can be given as a suitable non-negative number, *i.e.* the average rate, for transmitting the first window. Assume that the current window is $f_i f_{i+1} \ldots f_{i+W-1}$ (W is the window size) and the media frames $f_j f_{j+1} \ldots f_{i+W-1}$ are not transmitted as shown in Fig. 4. For the transmission segment $<p_i, p_{i+1}, r_i>$, the transmission starts from time t_i*T_f and ends at time $t_{i+1}*T_f$ with the bandwidth (transmission rate) r_i. In this paper, we simply define the segment point p_i as the lower corner of $H(t_i)$ ($H(t_i)^-$) or the upper corner of $F(t_i)$. It is not difficult to extend the segment points to any time points as shown in [6]. As shown in Fig. 4, there are three transmission segments $<p_0, p_1, r_0>$, $<p_1, p_2, r_1>$ and $<p_2, p_3, r_2>$. Notably, in the conventional approaches, r_2 is usually much smaller than r_{max} as the bandwidths in these two window are minimized independently. Thus, the large frames would not be pre-transmitted to reduce the peak bandwidth allocated in the next window. In our proposed approach, we try to construct a new transmission schedule to further utilize the peak bandwidth allocated in the previous windows. Based on this threshold parameter r_{max}, we do not only smooth the traffic but also pre-transmit the burst traffic as early as possible to minimize the required bandwidth in the next window.

3.1. Proposed Algorithm to Minimize Required Bandwidth

Before describing the proposed algorithm, we first define (s, $G(s)$) as the start-point of the segment and the increasing variable t is tested as the possible end-point of the segment. Assume that the media frames in the current window are $f_i f_{i+1} \ldots f_{i+W-1}$. In this paper, we define the functions $R_F(t) = (F(t) – G(s) - Q) / (t-s)$ and $R_H(t) = (H(t)^- – G(s) - Q) / (t-s)$ as the test rates that started from time s and ended at time t for the curves $F(x)$ and $H(x)$, respectively. $Q = F(j*T_f)^- - F(i*T_f)$ is the pre-transmitted traffic size by the previous transmission schedule. The detail description of the proposed algorithm in each window is shown as follows.

```
ALGORITHM: Online Traffic Smoothing
$s = i*T_f$, $t = t_H = t_F = j*T_f$, and $G(s) = F(j*T_f)^-$          // Initial the segment points
$r_F = R_F(s)$ and $r_H = R_H(s)$                                       // Initialize the transmission rates
repeat
    $t' = t + T_f$
    if ($r_H < R_F(t')$) {        // up-bounded by $H(x)$
        output segment: < $G(s)$, $H(t_H)^-$, $r_H$ >
        $r_{max}$ = max{ $r_{max}$, $r_H$ }, $G(t_H) = H(t_H)^-$, $Q=0$, $s = t_H$, $t = s + T_f$, $r_F = R_F(s)$ and $r_H = R_H(s)$
    } else if ($r_F > R_H(t')$) {        // low-bounded by $F(x)$
        output segment: < $G(s)$, $F(t_F)$, $r_F$ >
        $r_{max}$ = max{ $r_{max}$, $r_F$ }, $G(t_F) = F(t_F)$, $Q=0$, $s= t_F$, $t = s + T_f$, $r_F = R_F(s)$ and $r_H = R_H(s)$
    } else if ($t' = i+W-1$) {        // final open area
        if ($r_H < r_{max}$) {
            output segment: $< G(s), H(t_H)^-, r_H >$
            $G(t_H) = H(t_H)^-, Q=0, s = t_H, t = s + T_f, r_F = R_F(s)$ and $r_H = R_H(s)$
        } else {      // up-bounded by $r_{max}$
            $r_{max} = \max\{ r_{max}, r_F \}, t_F = \lceil (F(i+W-1)-G(s)) / (r_{max}*T_f) \rceil *T_f$
            output segment: $< G(s), F(i+W-1), r_{max} >$
            $s = t'$
        }
    } else {      // try the next frame
        $t = t'$
        if ($r_H \leq R_H(t)$) { $r_H = R_H(t), t_H = t$ }
        if ($r_F \leq R_F(t)$) { $r_F = R_F(t), t_F = t$ }
    }
until ($s = i+W-1$)
```

As suggested in [21], we choose the window size W to be an integer multiple of the size of GOP (group-of-picture). In this paper, we assign W as the number of frames in a GOP. Notably, if the smoothing window is started by a large I-frame, the required peak bandwidth would be highly increased. In this paper, whenever the remainder of the media stream is started by an I-frame, we automatically adjust the window size to resolve this problem. For example, we can extend the window size to $W+1$ to pre-transmit this I-frame in the current window. In online transmission, we usually assign the value of d as one frame.

Notably, in conventional window-based approaches, the number of sliding-frames is a constant value $k \leq W$. Given the sliding size k, the traffic-smoothing procedure will be executed periodically after k new frames are generated as shown in Fig. 5. Although the small sliding size could lead to the small peak bandwidth, the required computation time would be too high to online processing. Besides, it is hard to decide the suitable sliding size to minimize both the computation time and the peak bandwidth allocated. In this paper, we automatically decide the suitable sliding size. Our proposed approach executes the smoothing procedure only when the previous scheduled frames are all transmitted. Notably, the sliding size that considered in each window might be different with different numbers of frames (see Fig. 5). Generally, the applied window is hopping (sliding size = W) for CBR media traffic. When the burst traffic is presented, the siding size will be automatically decreased to make a better control. Thus, the peak bandwidth requirement can be minimized by pre-transmitting large frames.

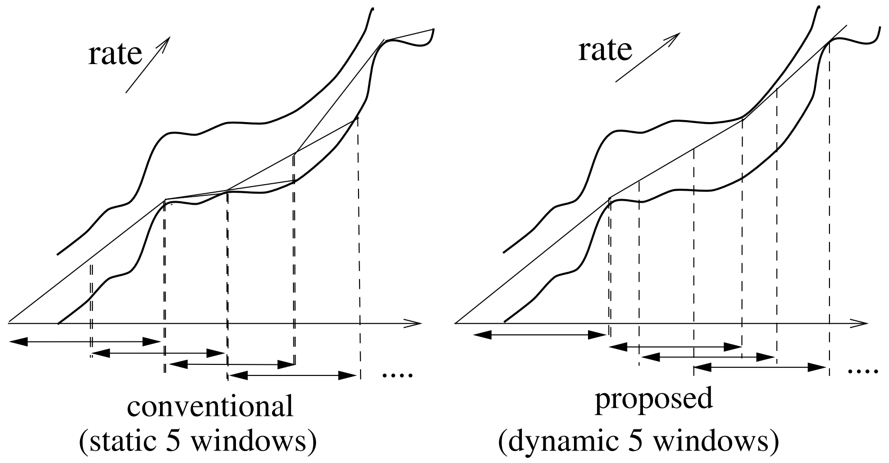


Fig. 5. In conventional approaches, the window sliding size is a given constant. In our approach, we can automatically decide the suitable sliding size to shape the VBR traffic. The traffics between different windows are also smoothed.

3.2. Bounded Bandwidth Allocation in Real-World Applications

In real-world applications, the online media stream may be latency- or quality-tolerance with a bounded delay in playback and a bounded decay in quality. They may have a bounded network bandwidth and try to admit as many requests as possible. Whenever the network congestion is detected and the available bandwidth is decreased, a feedback control algorithm could be applied to slow-down the encoding speed and decreases the encoding quality to reduce the media traffic. In conventional approaches, they simply assume that the media frames do not dropped or deferred during transmission. However, these assumptions are not guaranteed in current networks. To resolve this bounded bandwidth allocation problem, we can extend the feedback control algorithm to reduce the value of r_{max} to design a new transmission schedule. The modified $r_{max} = \min\{ r_{available}, r_{max} \}$. Notably, if the available bandwidth is too small, some frames may miss their transmission deadlines for continuous playback. As shown in Fig. 6, given the available bandwidth, the transmission deadline for each media frame can be easily computed by a linear-time algorithm based on the algorithms proposed in [2]. As shown in Appendix A, this algorithm can be proved to be optimal in the utilizations of allocated buffer and bandwidth. Considering a MPEG stream, we can directly drop P- and B-frames if their deadlines are missed. If an I-frame has missed its transmission deadline, we can try to drop other B/P-frames or to defer the stream display as shown in Fig. 6. Thus, the modified media traffic can fit the available bandwidth.

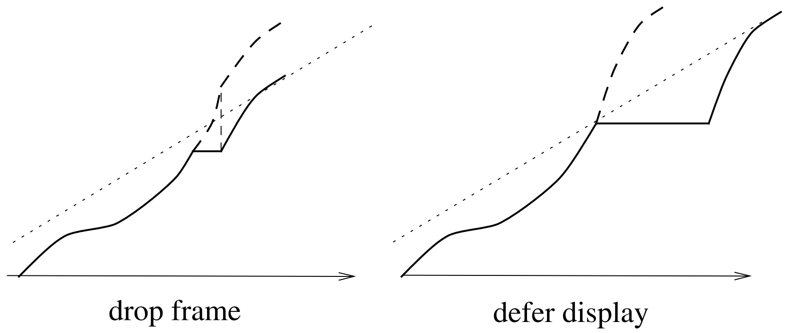


Fig. 6. If the available bandwidth is too small, some frames may miss their transmission deadlines and the buffer may be underflow. We can drop frames or defer the stream display to reduce the current media traffic to fit the allocated bandwidth by a feedback controller.

Table 1. Statistics of the test VBR-encoded MPEG media streams. (fps: frame-per-second, AVG: average bit-rate, STD: standard deviation of frame size)

Stream	Frame #	fps	Max Frame	AVG	STD	Burst Traffic
Star War	174136	24	22.62 KB	0.36 Mb/s	2.3	Convex-and-Concave
Princess Bride	167766	30	29.73 KB	1.17 Mb/s	4.8	None (large STD)
CNN News	164748	30	30.11 KB	1.17 Mb/s	3.7	None
Lecture	16316	30	6.14 KB	0.33 Mb/s	1.6	Convex
Advertisements	16316	30	10.08 KB	0.45 Mb/s	1.9	Convex-and-Concave

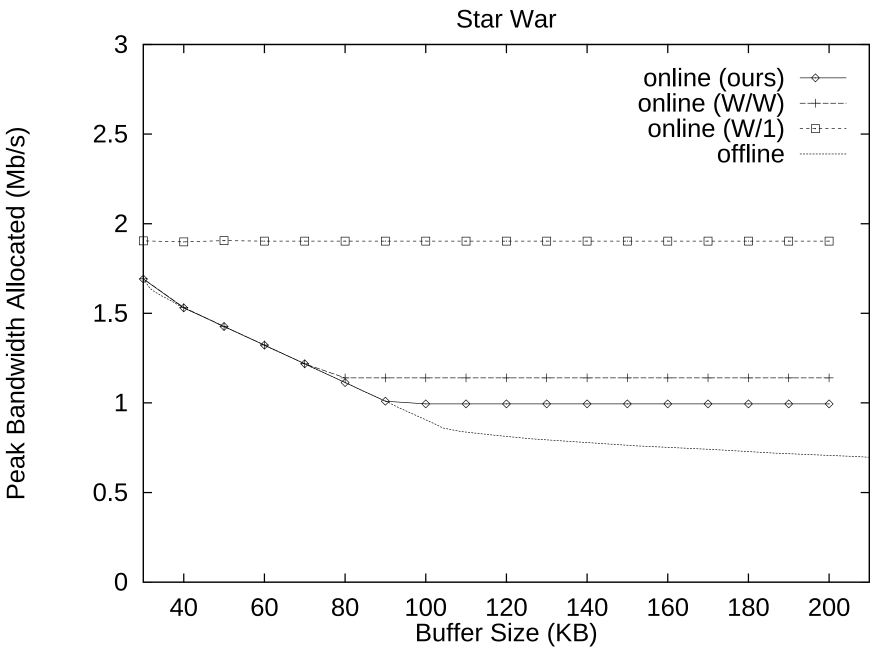
4. Experiment Results

In this paper, we have examined the proposed online transmission method by many benchmark streams [25-26]. Without losing the generalization, we evaluate the proposed approach by the peak bandwidth allocated and the network idle rate (the bandwidth utiliization – the actual bandwidth consumed divides the peak bandwidth allocated). Comparisons are made with an optimal offline scheduler proposed in [2] and the conventional window-based online schedulers presented in [21]. To consider the long-length movie stream, our first test example is an over two hours long MPEG-encoded *Star War* movie. In *Star War*, the number of frames in a GOP (called *frame rate*) is 24 fps (frames-per-second). It has large media frames and high variation in frame sizes as many real-world media streams. To test the video stream encoded by hardware MPEG coder, the next two media traces examined are nearly 90 minutes long *CNN News* and *Princess Bride* video streams. As the hardware coder uses variable distortion coding to maintain its target rate, the obtained variation in frame sizes is small. The last two test data are *Advertisements* and *Lecture* video traces encoded by the UCB software MPEG coder [25] with constant distortion coding. Their variations in frame sizes are different due to the video contents are different. In *Lecture*, the same speaker and his slides are shown along with only zooming and panning. As the frame contents are very similar, the variation in frame sizes is small. However, in

Advertisements, the frame contents are changed from one scene to another scene in a sequence of advertisements. Although the mean value of frame sizes is small, the frame size variation is very high. Table 1 shows the detail statistics of test media streams.

4.1. Comparisons with Conventional Approaches

Notably, as the window sliding size used in our dynamic-sliding algorithm would be larger or equal to that of SLWIN(1), it can be easily proved that our proposed approach would require smaller computation time than that of SLWIN(1) (the one-frame sliding-window approach). Although we have introduced a fast algorithm to speedup SLWIN(1) by computing only the first transmission segment in each window, the consumed CPU time is still very large. Our experiments show that the proposed approach can compute almost two times faster than the fast SLWIN(1) algorithm. When comparing the required bandwidth and network idle rate, experiments to the *Star War* movie show that the required bandwidth of SLWIN(1) is over 13% higher than that of our proposed approach (as shown in Fig. 7). Besides, the obtained network idle rate of SLWIN(1) is over 4% higher than that of our proposed approach. Our proposed approach is shown efficient and effective. Although the SLWIN(*W*) approach with a *W*-frame hopping-window could compute fast, the required bandwidth and the obtained network idle rate are too high. The requirements in large system resources make conventional approaches impractical to play back burst VBR media streams such as the *Star War* movie. Fig. 8 shows the required network bandwidth and idle rate to online transmit the *Advertisements* video stream. Comparisons are made with the optimal offline scheduler (*offline*), the SLWIN(1) approach (*online (W/W)*) and the SLWIN(*W*) approach (*online (W/1)*). Our obtained result is better than that of the conventional SLWIN(1) approach.



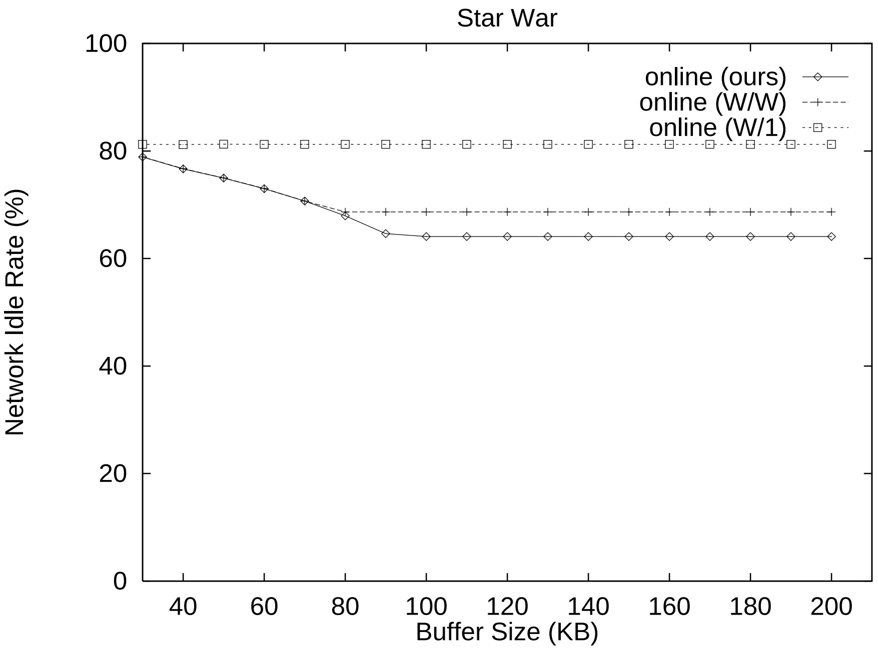
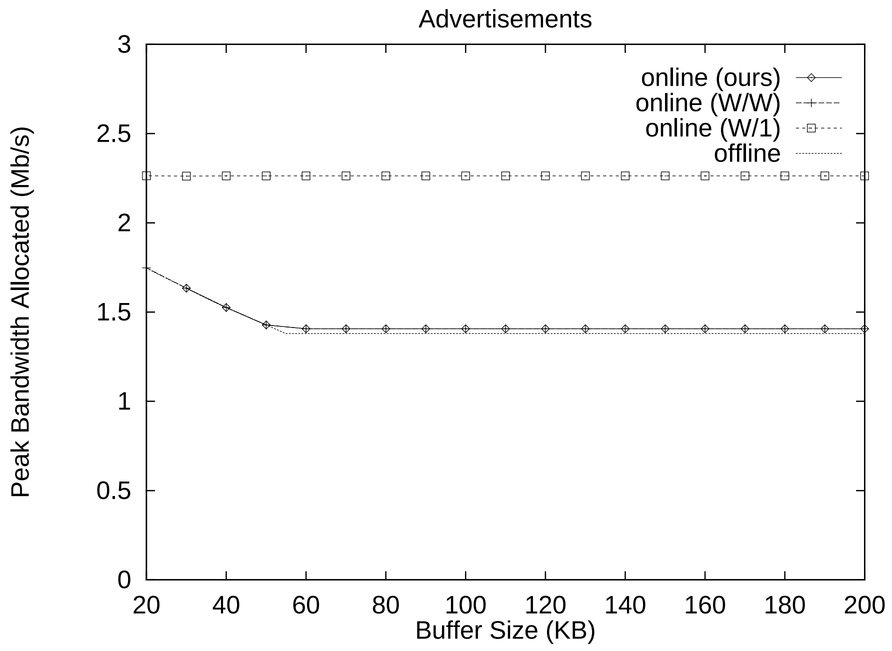


Fig. 7. The required network bandwidth and idle rate to online transmit the *Star War* movie stream. Comparisons are made with the optimal offline scheduler (*offline*), the SLWIN(1) approach (*online (W/W)*) and the SLWIN(*W*) approach (*online (W/1)*).



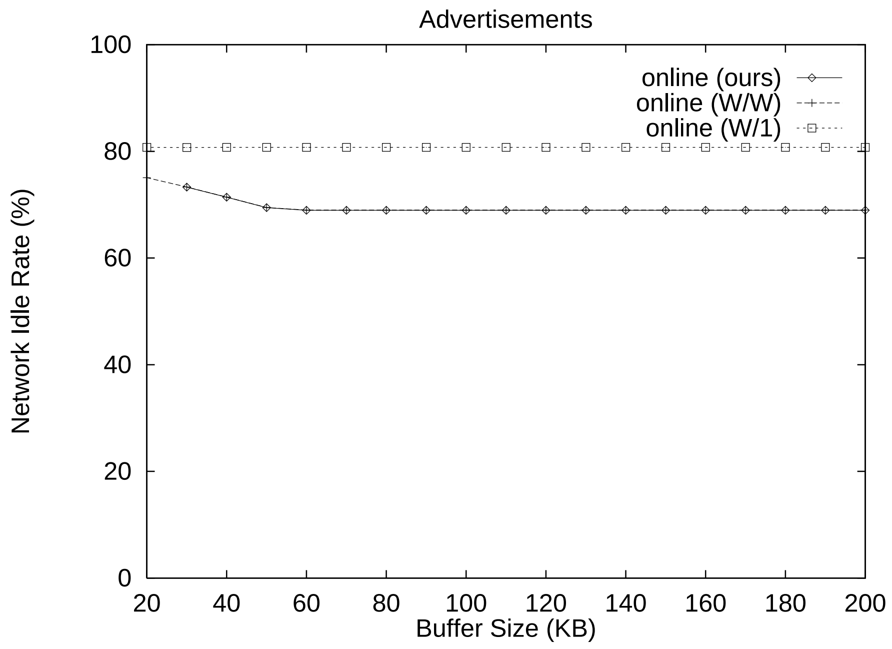


Fig. 8. The required network bandwidth and idle rate to online transmit the *Advertisements* video. Comparisons are made with the optimal offline scheduler (*offline*), the SLWIN(1) approach (*online (W/W)*) and the SLWIN(*W*) approach (*online (W/1)*).

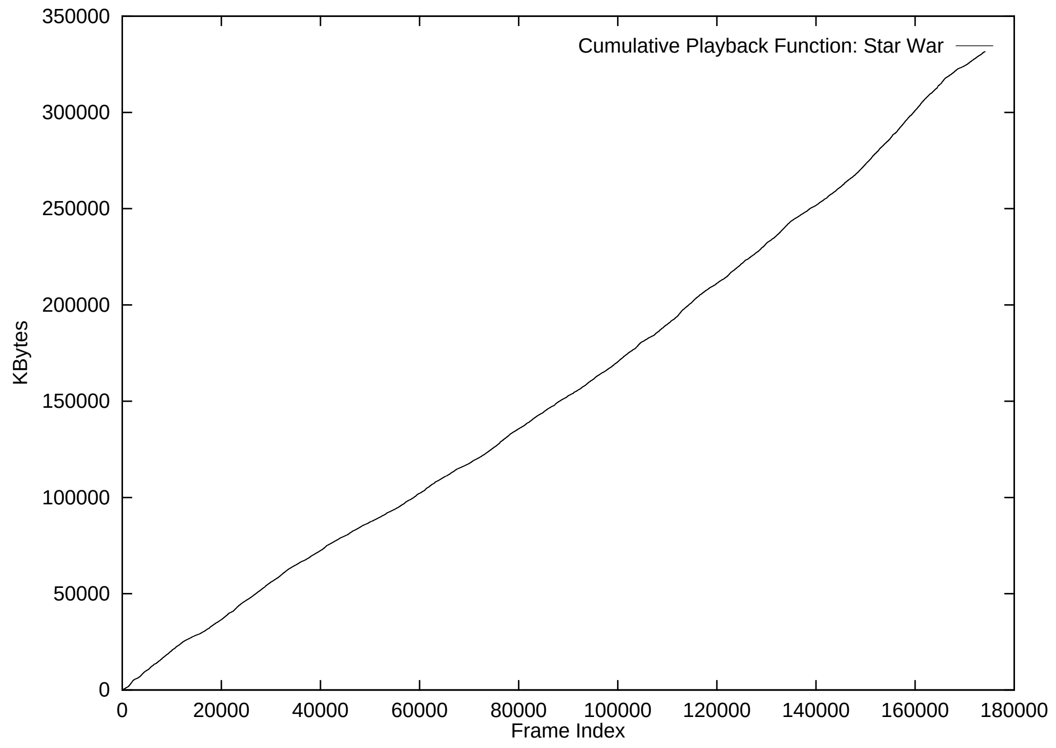


Fig. 9. The cumulative playback function of the *Star War* movie stream. This media stream contains a long-term concave burstiness.

4.2. Obtained Results for Different Types of Burst Traffics

In this paper, to take a more in-depth look at the proposed algorithm, we demonstrate how the concave-covex behaviors of input media can influence the required system resources in transmission. Our experiments show that, given a suitable buffer size (> $\max\{f_i\}$), the required bandwidth depends primarily on the characteristics of stream burstiness. If the video stream has a sustained area of small frames followed by a sustained area of large frames, it contains a *concave* burstiness. This kind of burstiness is hard to predict by the previous traffic size. The required bandwidth for online transmission tends to be higher than stream's average bit-rate even a large buffer size is allocated. For example, the *Star War* movie stream contains a long-term concave burstiness (see Fig. 9). As shown in Fig. 7, the increasing of buffer size does not lead to the decreasing of allocated network bandwidth when the buffer size is larger than 80 KB. Comparing to the conventional SLWIN(1) and SLWIN(*W*) approaches, our proposed approach can utilize the allocated network bandwidth more efficiently. Under the same constrains of initial delay and buffer size, our obtained network idle rate is smaller than those obtained by the conventional approaches. Furthermore, the obtained improvement is increased dramatically whenever a sufficient buffer is allocated.

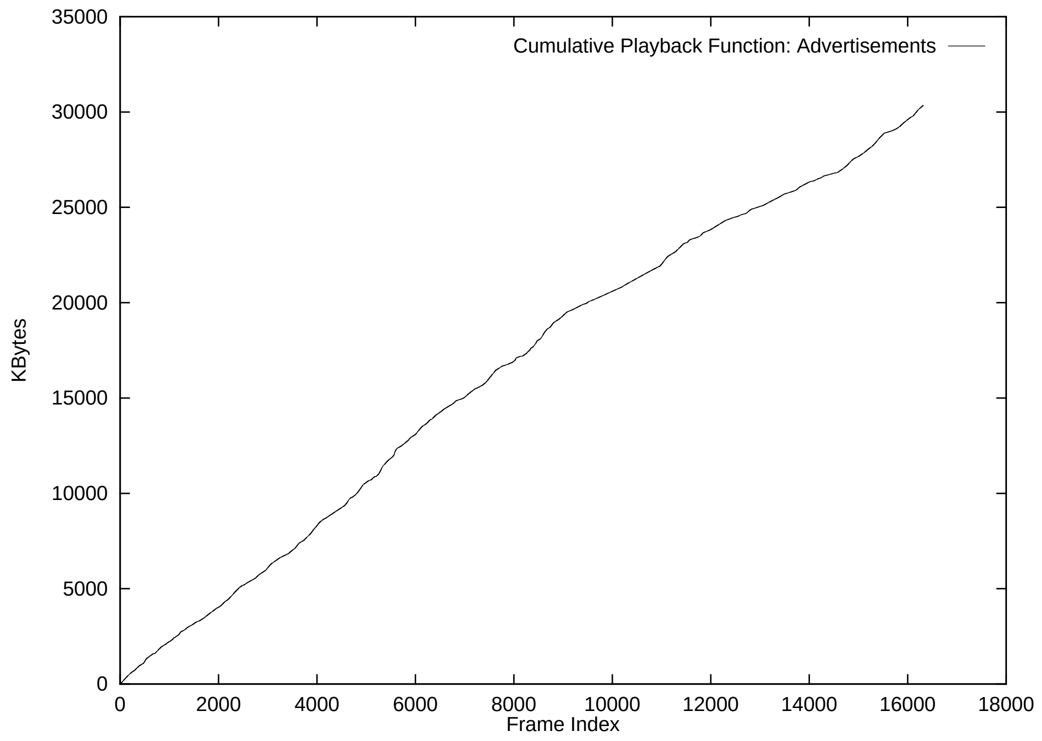


Fig. 10. The cumulative playback function of the *Advertisements* video stream. This media stream contains a long-term convex burstiness.

Different from the *concave* burstiness, a sustained area of larger frames may be followed by a sustained area of smaller frames. It presents a *convex* burstiness that can be found in the *Advertisements* video stream (see Fig. 10). With a *convex* burstiness, the burst traffic is usually presented at the first window and can be easily identified. As shown in Fig. 8, our allocated peak bandwidth is almost the same as that allocated by the SLWIN(1) approach. When comparing the obtained network idle rate, our proposed approach is better than the SLWIN(1) approach. Generally, a video stream may contain both the *concave* and *convex* long-term burstiness (as shown in *Star War* and *Advertisements*). Comparing the *Lecture* video that contains only the convex burstiness to the *Advertisements* video, the relations between the required bandwidth and the available buffer size for *Advertisements* is more complex than that for *Lecture*. The cumulative playback function of *Lecture* is shown in Fig. 11. The required peak bandwidth and the obtained network idle rate are shown in Fig. 12. We can find that these two video streams have the same frames number and the same GOP size. Besides, their average frame rates and variations in frame sizes are similar. The only difference between these two media streams is the characteristic of burstiness. This experiment shows that the obtained results of online traffic smoothing highly depend on the characteristic of stream burstiness.

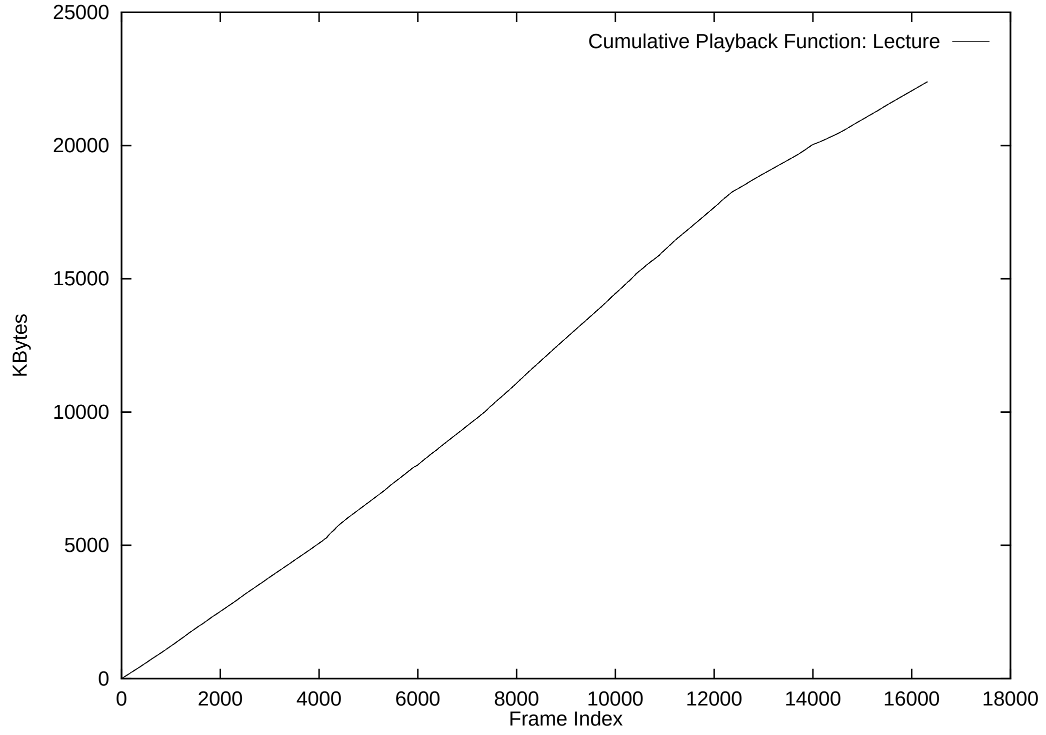
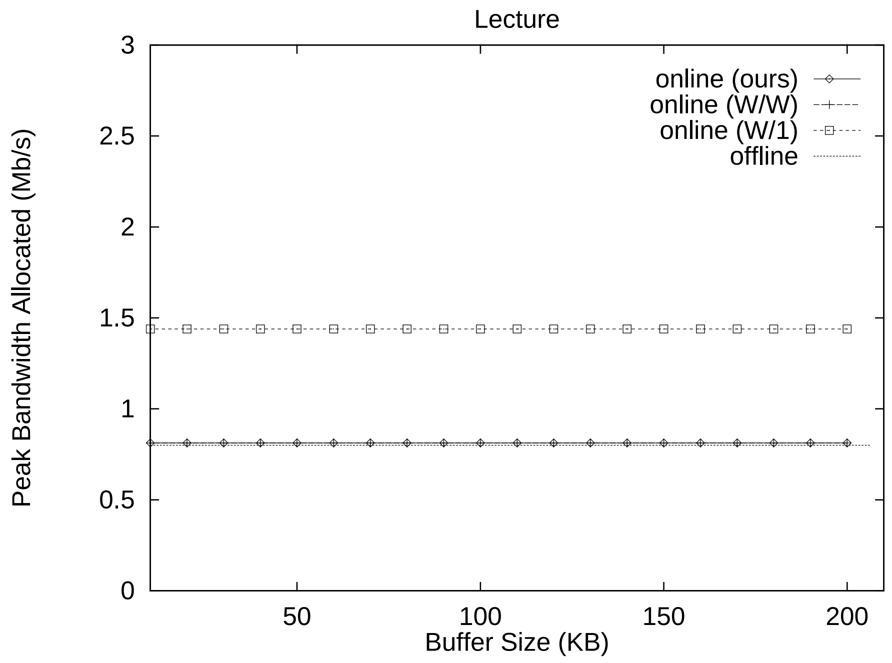


Fig. 11. The cumulative playback function of the *Lecture* video stream. This media stream contains only a long-term convex burstiness.



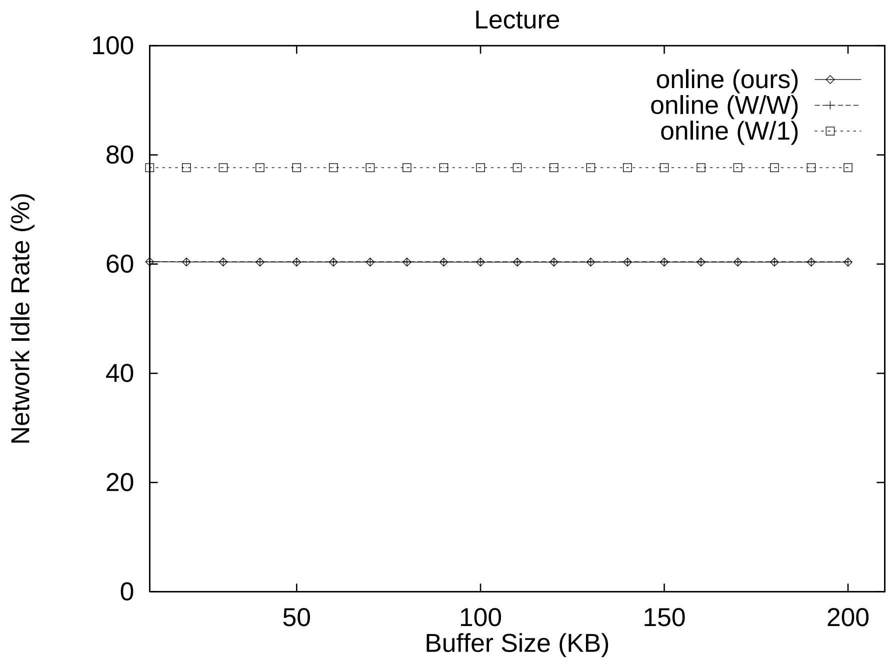


Fig.12. The required network bandwidth and idle rate to online transmit the *Lecture* video stream. Comparisons are made with the optimal offline scheduler (*offline*), the SLWIN(1) approach (*online (W/W)*) and the SLWIN(*W*) approach (*online (W/1)*).

4.3. Obtained Results for Different Frame Size Variations

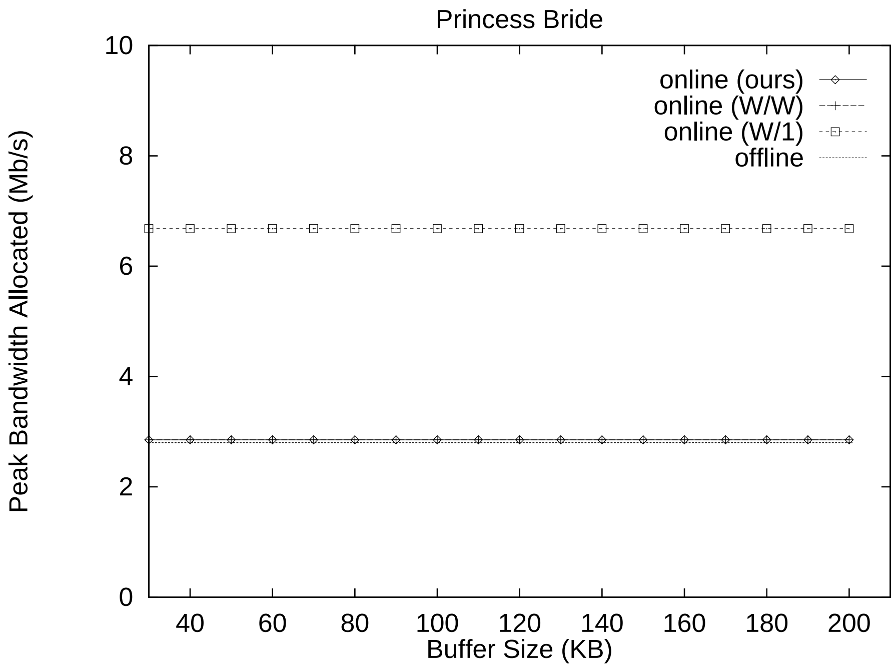
If a video stream has no *concave* or *convex* long-term burstiness, the required bandwidth would highly depend on the frame size variation. In this paper, we use the standard deviation (STD as show in Table 1) to measure the frame size variations for different media streams. Generally, the standard deviation for media stream $f_0 f_1 \ldots f_{n-1}$ (f_i is the i-th media frame size) is defined as follows.

$$\text{STD} = \text{sqrt}\{\ ((f_0 - \text{mean})^2 + (f_1 - \text{mean})^2 + \ldots + (f_{n-1} - \text{mean})^2) \mathbin{/} n\ \}$$

$$\text{where} \quad \text{mean} = (f_0 + f_1 + \ldots + f_{n-1}) \mathbin{/} n$$

Comparing *Princess Bride* and *CNN News*, these two media streams have the similar stream length and the same average frame rate. However, the frame size variation of *Princess Bride* is higher than that of *CNN News*. Comparison results of the required bandwidth and the obtained network idle rate are shown in Fig. 13 and Fig. 14. In our experiments, the required peak bandwidth for *Princess Bride* would be larger than that for *CNN News*. The large variation in frame size would lead to the large requirement in peak transmission bandwidth.

Comparing to the video streams which contain long-term burstiness, our experiments also show that only for a nearly-CBR media stream (such as *Princess Bride*, see Fig. 15), the conventional SLWIN(*W*) approach may be considered as a good choice because of its simplicity. However, the obtained network idle rate of SLWIN(*W*) is still very large. Notably, the allocated bandwidth of our online transmission scheduler is the same as that obtained by the optimal offline transmission scheduler. Our proposed approach is shown to be effective and efficient, and can allocate a small network bandwidth for online VBR stream transmission. The obtained results are better than those obtained by the conventional SLWIN(1) and SLWIN(*W*) approaches.



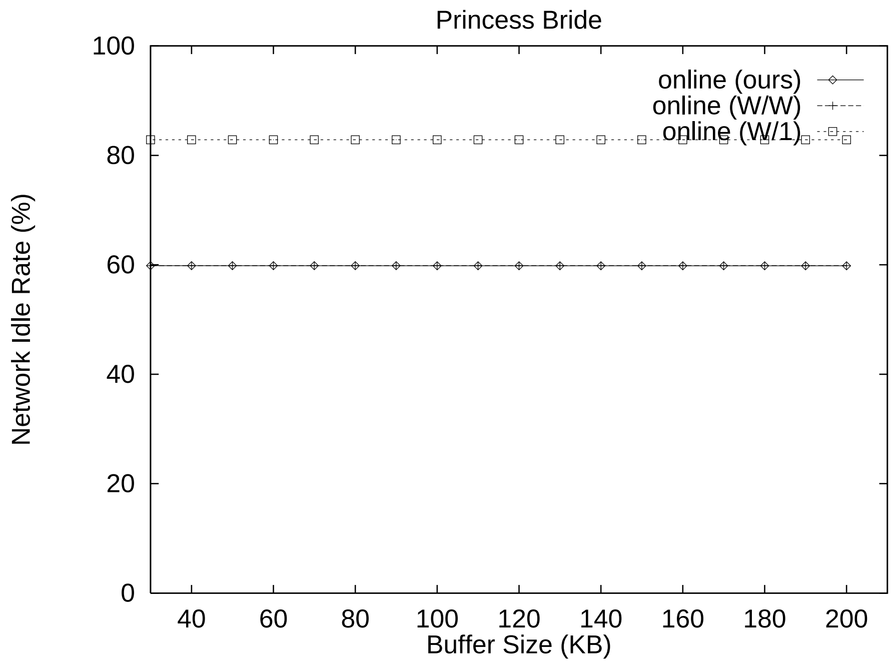
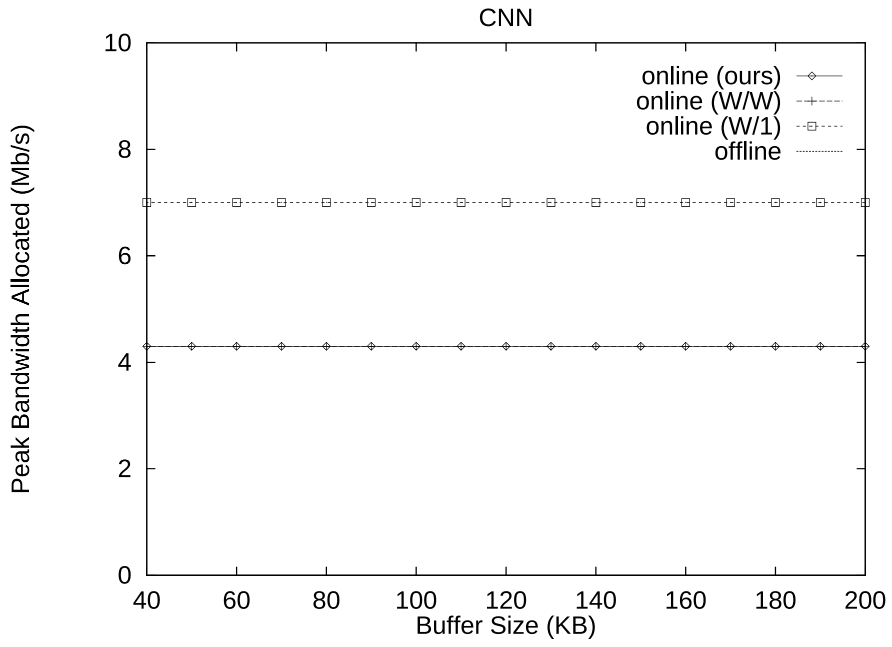


Fig.13. The required network bandwidth and idle rate to online transmit the nearly-CBR *Princess Bride* video stream. Comparisons are made with the optimal offline scheduler (*offline*), the SLWIN(1) approach (*online (W/W)*) and the SLWIN(*W*) approach (*online (W/1)*).



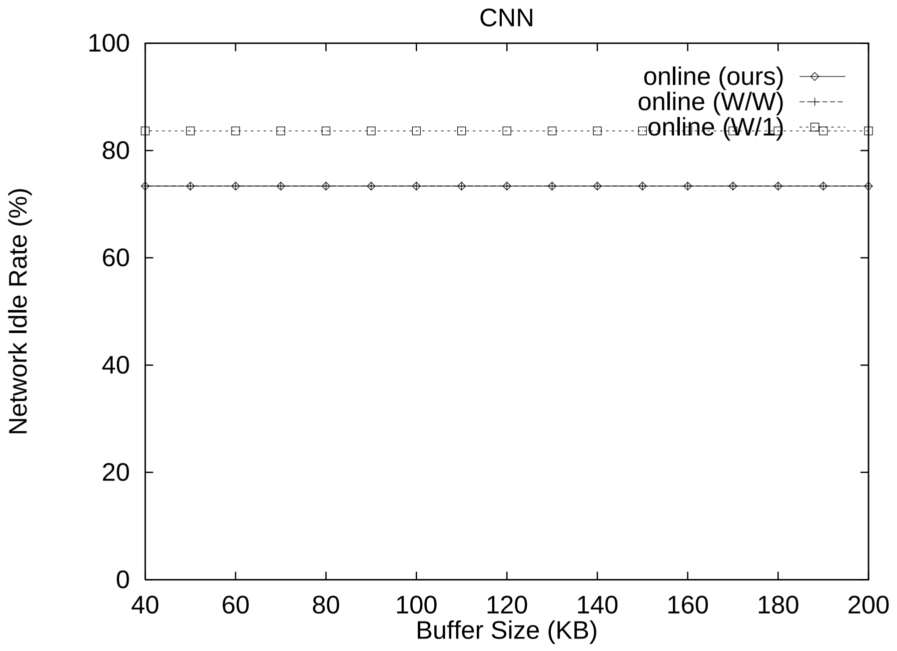


Fig. 14. The required network bandwidth and idle rate to online transmit the nearly-CBR *CNN News* video stream. Comparisons are made with the optimal offline scheduler (*offline*), the SLWIN(1) approach (*online (W/W)*) and the SLWIN(*W*) approach (*online (W/1)*).

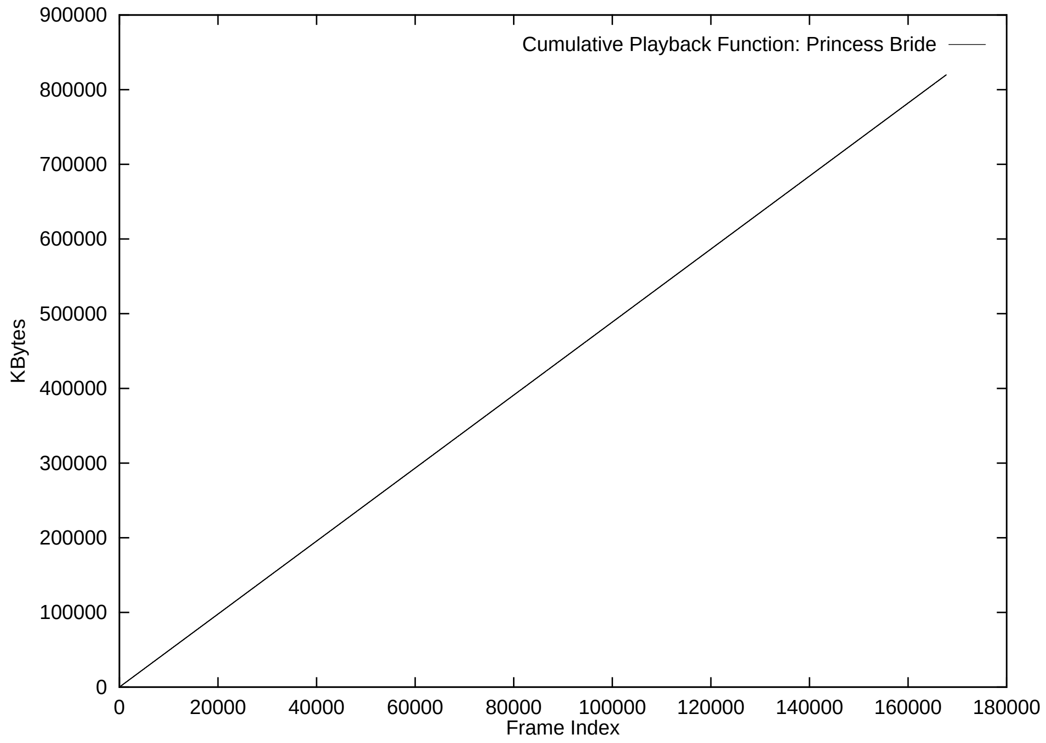


Fig. 15. The cumulative playback function of the *Princess Bride* video stream. It is a nearly-CBR media stream.

5. Conclusion

This paper presents a traffic smoothing method for delivery of online media stream. The proposed approach can automatically decide the suitable size of window sliding to smooth the VBR traffic under the minimum bandwidth requirement. We have explored the proposed algorithm by transmitting several benchmark video streams. Given the client buffer size and the playback delay, our proposed approach would require smaller peak bandwidth than that of conventional approach with 13% improvements when transmitting the *Star War* movie stream. Besides, the utilization of network obtained is also better than that obtained by the conventional approach. When considering the latency- and quality-tolerance applications, an encoder feedback control algorithm based our proposed algorithm and the algorithms proposed in [2] can be easily applied. Our proposed approach is shown effective and efficient.

Reference

[1] J.Y. Hui, J. Zhang and J. Li, "Quality of service control in GRAMS for ATM local area network," *IEEE Journal on Selected Areas in Communications*, May 1995.

[2] R. I Chang, M. Chen, M. T. Ko and J. M. Ho, "Optimizations of stored VBR video transmission on CBR channel," *Proc. SPIE VVDC: Performance and Control of Network Systems*, pp. 382-392, 1997.

[3] R. I Chang, M. Chen, M. T. Ko and J. M. Ho, "Designing the ON-OFF CBR Transmission Schedule for Jitter-Free VBR Media Playback in Real-Time Networks," *IEEE Int. Workshop on Real-Time Computing Systems and Applications*, pp. 1-9, 1997.

[4] J. M. McManus and K. W. Ross, "Dynamic programming methodology for managing prerecorded VBR sources in packet-switched networks," *Proc. SPIE Int. Symp. Voice, Video, and Data Communication: Performance and Control of Network Systems*, 1997.

[5] J. D. Salehi, Z. L. Zhang, J. F. Kurose and D. Towsley, "Supporting stored video: reducing rate variability and end-to-end resource requirements through optimal smoothing," *Proc. ACM SIGMETRICS*, pp.222-231, 1996.

[6] W. Feng and S. Sechrest, "Smoothing and buffering for delivery of prerecorded compressed video," *IS&T/SPIE Multimedia Computing and Networking*, pp. 234-242, February 1995.

[7] W. Feng, F. Jahanian and S. Sechrest, "Optimal buffering for the delivery of compressed prerecorded video," *IASTED International Conference on Networks*, Jan 1996.

[8] W. Feng, "Rate-constrained bandwidth smoothing for the delivery of stored video," *IS&T/SPIE Multimedia Computing and Networking*, pp. 316-327, 1997.

[9] J. M. McManus and K. W. Ross, "Video On Demand over ATM: Constant-Rate Transmission and Transport," *Proc. IEEE INFOCOM*, March 1996.

[10] M. Grossglauser, S. Keshav and D. Tse, "RCBR: a simple and efficient service for multiple time-scale traffic," *Proc. ACM SIGCOMM*, August 1995.

[11] H. Zhang and E. W. Knightly, "A new approach to support delay-sensitive VBR video in packet-switched networks," *Proc. 5th Workshop on Network and Operating Systems Support for Digital Audio and Video*, 1995.

[12] E. W. Knightly and D. E. Wrege and J. Liebeherr and H. Zhang, "Fundamental limits and tradeoffs of providing deterministic guarantees to VBR video traffic," *Proc. ACM SIGMETRICS*, pp. 47-55, 1995.

[13] S. Radhakrishnan and S.V. Raghavan, "A flexible traffic shaper for high-speed networks: design and comparative study with leaky bucket," *Tech. Report CS-TR-3495*, Inst. for Advanced Computer Studies, Univ. of Maryland, 1995.

[14] E.W. Knightly and P. Rossaro, "Smoothing and multiplexing tradeoffs for deterministic performance guarantees to VBR video," *Proc. Int. Symp. Multimedia Communication and Video Coding*, 1995.

[15] J. Rexford and D. Towsley, "Smoothing variable-bit-rate video in an internetwork," *Proc. SPIE Int. Symp. Voice, Video, and Data Communication: Performance and Control of Network Systems*, 1997.

[16] A. R. Reibman and A. W. Berger, "Traffic descriptors for VBR video teleconferencing over ATM networks," *IEEE/ACM Transactions on Networking*, June 1995.

[17] M. Grossglauser and S. Keshav, "On CBR Service," Proc. IEEE INFOCOM, March 1996.

[18] Ming-Tat Ko, Jan-Ming Ho, Meng Chang Chen and Ray-I Chang, "Characterize the minimum required resources for admission control of pre-recorded VBR video transmission by an O(N log N) algorithm," submitted for publication, 1998.

[19] S. S. Lam, S. Chow and D. K. Y. Yau, "An algorithm for lossless smoothing of MPEG video," *Proc. ACM SIGCOMM*, 1994.

[20] T. Ott, T. V. Lakshman and A. Tabatabai, "A scheme for smoothing delay-sensitive traffic offered to ATM networks," *Proc. IEEE INFOCOM*, 1992.

[21] J. Rexford, S. Sen, W. Feng, K. Kurose, J. Stankovic and D. Towsley, “Online Smoothing of Live, Varibale-Bit-Rate Video,” *Proc. Workshop on Network and Operating Systems Support for Digital Audio and Video*, 1997.

[22] M. Chen, D. D. Kandlur and P. S. Yu, “Optimization of the grouped sweeping scheduling (GSS) with heterogeneous multimedia streams,” *ACM Multimedia Conference*, pp. 235-142, 1993.

[23] E. Chang and A. Zakhor, “Scalable video data placement on parallel disk arrays,” *IS&T/SPIE Symposium on Electronic Imaging Science and Technology*, February 1994.

[24] Y.C. Wang, S.L. Tsao, R.I. Chang, M.C. Chen, J.M. Ho and M.T. Ko, “A fast data placement scheme for video server with zoned-disks,” *Proc. SPIE VVDC: Multimedia Storage and Archiving System II*, pp. 92-102, 1997.

[25] http::/www.eeb.ele.tue.nl/mpeg/

[26] ftp::/thumper.bellcore.com

[27] J. Zhang, “Optimal buffering algorithms for client-server VBR video retrievals,” Ph.D. Thesis, Computer Science Department, The State University of New Jersey, 1997.

[28] R.I. Chang, W.K. Shih and R.C. Chang, "A new real-time disk scheduling algorithm and its application to multimedia systems," in *Lecture Notes on Computer Science: 5th IEEE IDMS*, 1998.

Appendix: A

ALGORITHM: identify the frames those miss their transmission deadlines

input: video stream *V*, the peak transmission rate *r*, the client buffer *b* and the initial delay *d*

output: the media frames those miss their transmission deadline

```
$A_{-1}$ = min{ $|V|$, $b$, $(d-1)*r$ }
for $i = 0$ to $n-1$ do {
    $A_i$ = min{ $|V|$, $F_{i-1} + b$, $A_{i-1} + r*T_f$ } // $F$ is the CPF
    if ($A_i < F_i$) {    // the frame $i$ will miss its transmission deadline
        the solution procedure (i.e. drop the frame or defer the frame playback time)
    }
}
```