

Practical Considerations for Smoothing Multimedia Traffic Transported over Packet-Switched Networks

Christos Tryfonas

Sprint Advanced Technology Laboratories
1 Adrian Court
Burlingame, CA 94010
tryfonas@sprintlabs.com

Abstract

The transport of multimedia streams over packet-switched networks often involves a preprocessing to smooth the variable bit-rate nature of the streams to achieve higher network utilization. By examining the smoothing procedure and its effects on an end-to-end basis, we identify critical points that should be taken into account by any smoothing algorithm. Such issues include the packet delay variation (jitter) throughout the network, the consideration of any unused data present in the stream, the clock recovery problem, and the use of alternate schedules in case of a renegotiation failure. We propose and discuss specific solutions for all the issues.

1 Introduction

A significant portion of the forecasted network traffic is expected to be multimedia (e.g. voice and video) traffic. New services such as video-on-demand (VoD) and TV broadcasting are currently under deployment. The video traffic usually exhibits high variability in its bandwidth demands in different time scales. In video applications that transport stored video over a packet-switched network the video stream is often smoothed by work-ahead smoothing, i.e., sending more data to the receiver with respect to its playback time. Significant work can be found in the literature in the area of work-ahead video smoothing [1, 6, 8, 10]. The general idea behind most of these algorithms is to maximize the time intervals (rate segments) at which a transmission rate for the video stream is used without causing under/overflow of the receiver buffer. The algorithms differ in the selection of the starting point of these rate segments.

All the known smoothing algorithms found in the literature generate an optimal transmission schedule (according to some optimization criterion) of a sequence of video frames by assuming a zero or a constant end-to-end delay through the packet-switched network. However, the end-to-end delay is often a function of the reserved network resources and therefore, the computed transmission schedule assuming a maximum *end-to-end delay variation (jitter)* is often conservative. In addition, the smoothing algorithms assume that all the data sent over the network are used by the decoder on the receiver side. However, a video stream often contains data which may not be relevant to the data to be decoded on the receiver side. This data needs not be inserted in the decoder buffer and therefore it should not be considered in any computation involving the decoder buffer. Thus, the transmission schedules computed by the existing smoothing algorithms cannot guarantee that the receiver buffer will not under/overflow.

Another issue not addressed in the literature is the impact of a smoothed video schedule to the clock recovery process on the receiver side. Clock recovery is needed in applications in which the sender and the receiver need to be synchronized such as in TV broadcasting. The reconstructed clock can be used in different functions such as for estimating the packet delay through the network, or synthesizing a chroma sub-carrier for the composite video signal of the TV set in the case of broadcasting applications. The use of applications with stringent clock specifications requires a careful selection of the video stream's delivery process. The transmission schedule for a given video stream as computed by the existing smoothing algorithms found in the literature [1, 6, 8, 10] forces the receiver buffer to almost underflow after almost overflowing and vice

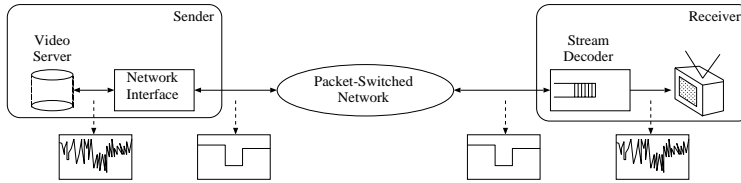


Figure 1: Abstract model to describe the smoothing procedure.

versa. This hardens the task of estimating the average rate of the stream, or reconstructing a stable clock from the incoming video stream.

Although an optimal transmission schedule can be computed, it may not be feasible to be followed at all times due to a renegotiation failure. This comes from the fact that a renegotiated service is often statistical with no hard guarantees. Two possible solutions have been proposed [3, 5]: (i) a dynamic requantization of the video stream so that the resulting stream can be sent with lower transmission rate, or (ii) use of a scalable encoding process in which a base layer is combined with a set of enhancement layers, so that the sender can decide to drop specific enhancement layers to lower the actual network resource demand and, therefore, the transmission rate of the stream. We propose another possible solution which makes use of a set of precomputed transmission schedules with convergence points. On a renegotiation failure, an alternate schedule may be followed until the convergence point between the old and the alternate schedules is reached at which the control is returned to the old schedule.

This paper addresses the problems described above in detail and suggests possible solutions in each case. The rest of the paper is as follows: In Section 2, we present the feasibility conditions that any transmission schedule of a video stream should follow so that receiver buffer under/overflow does not occur. In Section 3, we first derive the conditions corresponding to the computation of the transmission schedule in the case of variable network jitter, and then present a variation of the dynamic programming algorithm proposed by Jiang et al. [6] to compute an optimal transmission schedule for the case of variable network jitter. In Section 4, we show the impact of data present in the video stream that do not enter the decoder buffer on the transmission schedule decision, and provide solutions for both the elementary video stream and the MPEG-2 Transport Stream cases. In Section 5, we discuss possible effects of the transmission schedule on the clock recovery process at the decoder end in the case that the decoder clock is derived from the incoming stream, and give possible solutions. In Section 6, we assume a valid precomputed transmission schedule and discuss possible ways that the actual transmission schedule remains feasible even in the case that a specific renegotiation request cannot be granted. Finally, in Section 7 we conclude the paper with a summary of the points raised throughout this work.

2 Feasibility Conditions

The abstract model used to describe the smoothing procedure is shown in Figure 1. The video stream is either displayed locally or sent from the video server through a packet-switched network to a decoder with a buffer size of B_D bits. The network interface smooths the incoming video stream to facilitate the admission control process and provide higher utilization in the network.

We derive the feasibility conditions for both elementary video streams and MPEG-2 Transport Streams so that the transmission schedule does not cause any under/overflows of the receiver buffer. The feasibility conditions are derived for the case of variable end-to-end network delay. We start with the description of the notation used by the feasibility conditions. The notation is introduced by means of a *nominal case*, i.e., when the stream is displayed locally with the additional assumption that no delay variation is present in the data delivery process.

2.1 Notations for the Nominal Case

In the case that the video stream is not sent over a packet network and it is displayed locally, the corresponding buffer dynamics constitute the nominal case. First, we present the notation for the case of elementary

video streams. We then introduce the corresponding notation for the case of MPEG-2 Transport Streams.

2.1.1 The Elementary Video Stream Case

In the case of an elementary video stream displayed locally, we define the function $C_N(t)$ as the cumulative data consumed by the decoder at time t given by

$$C_N(t) = \sum_{i=0}^k s_i, \quad k = \lfloor ft \rfloor, \quad (1)$$

where s_i is the size of the i -th video frame of an elementary video stream with a frame rate of f frames/second. By assuming that all the bits of a frame are generated and sent to the decoder instantly, it becomes evident from Eq. (1) that $C_N(t)$ has discontinuities at the frame boundaries. Therefore, it can be considered discrete in the time domain at the time instants that frames have to be consumed (every $1/f$ second).

If frame-reordering is needed at the decoder, then we need to define $R_N(t)$ as the cumulative function of the minimum amount of data that should be present at the decoder at time t for correct operation, i.e.,

$$R_N(t) = \sum_{i=0}^k s_i, \quad k \geq \lfloor ft \rfloor, \quad (2)$$

Clearly, if no reordering is needed, the two functions defined above coincide, i.e., $R_N(t) = C_N(t)$.

2.1.2 The MPEG-2 Transport Stream Case

When an MPEG-2 Transport Stream is considered, the internals of the stream have to be taken into account for the nominal case. If the schedule implied by the structure of the MPEG-2 Transport Stream is followed by the decoder (use of *Program Clock Reference* (PCR), *Presentation* (PTS) and *Decoding Timestamps* (DTS)), then the system decoder buffer dynamics are available in advance for a specific stream. Following the terminology introduced in the elementary video stream case, we define as $R_N(t)$ the cumulative function that records all the data received up to time t given by

$$R_N(t) = \int_0^t r_N(t) dt, \quad (3)$$

where $r_N(t)$ is the transport rate of the stream at time t in bits/sec.

We also define $c_N(t)$ as the function that is positive only at the time instants (corresponding to the associated PTS or DTS value) at which the access units [5] are consumed by the MPEG-2 system decoder buffer and forwarded to the elementary decoders. This function has the value of the size of the access unit to be forwarded at those time instants and zero at all others. The corresponding cumulative function $C_N(t)$ that records all the data consumed by the decoder up to time t is now given by

$$C_N(t) = \int_0^t c_N(t) dt. \quad (4)$$

Both $R_N(t)$ and $C_N(t)$ functions are shown in Figure 2 in which the buffer dynamics through time are plotted for the nominal case. It should be noted that $C_N(t)$ need not be continuous but discrete in the time domain, since access units may be forwarded only at time instants that are multiple of the period of a 90 kHz clock used for this purpose by the decoder [4].

Although both $R_N(t)$ and $C_N(t)$ functions are defined by implying that sender and receiver are synchronized (even when displayed locally), they can be obtained to reflect the initial fluctuations due to the clock acquisition process by sending the stream to the decoder and recording the buffer dynamics. However, in such a case, both functions will correspond to a specific clock acquisition process.

We continue with the derivation of the feasibility conditions that should be satisfied by the transmission schedule so that either an elementary or an MPEG-2 system decoder buffer with size B_D never underflows or overflows. These conditions can be derived for various network conditions: (i) constant end-to-end delay, (ii)

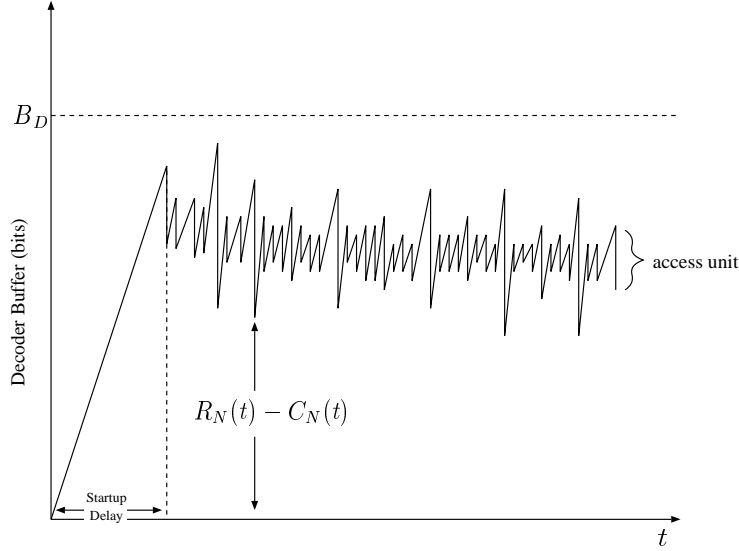


Figure 2: Buffer dynamics for the nominal case for a typical MPEG-2 Transport Stream.

worst-case end-to-end delay variation, and (iii) variable network delay dependent on the reserved network bandwidth. Due to space constraints, we obtain the conditions for the case of network delays dependent on the reserved bandwidth. The remaining cases can be treated as special cases. The interested reader is referred to [14] for in-depth discussion of all the cases.

2.2 Variable Network Delays Dependent on the Reserved Rate

There are cases in which the network delay is dependent on the reserved rate [2, 9, 12, 11, 15]. More specifically, the worst-case delay is usually a decreasing function of the reserved rate in the network. Let $R_D(t)$ denote the cumulative data received by the decoder at time t . Let also d_m be the minimum packet delay due to propagation and minimum processing delays through the network elements. The cumulative function $R_D(t)$ is bounded by two functions defined as $R_{Dm}(t)$ and $R_{DM}(t)$ respectively. To ensure that the decoder buffer never underflows or overflows the following conditions must hold

$$R_{DM}(t) \leq R_D(t) \leq R_{Dm}(t), \quad t \geq 0 \quad (5)$$

$$R_{DM}(t) \geq C_N(t - d_m), \quad t > d_m \quad (6)$$

$$R_{Dm}(t) \leq C_N(t - d_m) + B_D, \quad t > d_m \quad (7)$$

$$R_{Dm}(t), R_{DM}(t) = 0, \quad 0 \leq t \leq d_m \quad (8)$$

We now continue with guidelines for performing smoothing under the case of variable network jitter.

3 Smoothing under Variable Network Jitter

The smoothing algorithms found in the literature compute a transmission schedule for a video stream under the assumption that the packet delay variation (jitter) from the sender to the receiver is fixed. However, the maximum jitter is often dependent on the reserved rate of the connection carrying the video stream. Therefore, an optimal smoothing algorithm should also consider the variable nature of the jitter as a function of the reserved resources for computing an optimal transmission schedule.

3.1 Derivation of Bounding Functions

The functions that bound the cumulative rate function at the decoder $R_D(t)$ can be obtained systematically when the reserved rate is increased or decreased from its current value. Since the slope of these functions

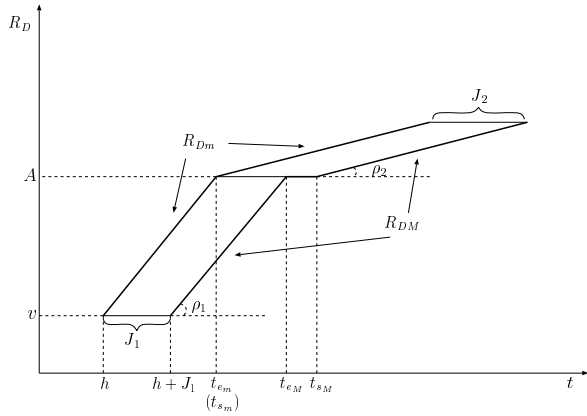


Figure 3: Example of a rate decrease ($\rho_2 < \rho_1$) and the resulting ambiguity on the rate received on the decoder side R_D .

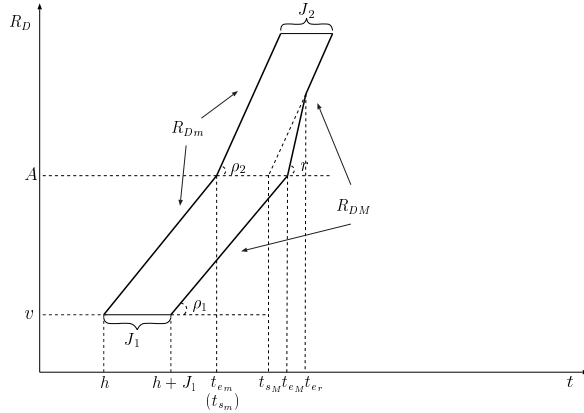


Figure 4: Example of a rate increase ($\rho_2 > \rho_1$) and the resulting ambiguity on the rate received on the decoder side R_D .

may only change when the reserved rate of the stream changes, we need to consider two special cases: in the first case, the sending rate is decreased from its previous value, whereas in the second case, the sending rate is increased. We derive the ambiguity zones for $R_D(t)$ under both cases.

3.1.1 The Case of Rate Decrease

Let us assume that ρ_1 is the sending rate before the decrease which becomes ρ_2 after the decrease. Clearly, the corresponding maximum delay is greater after the decrease, resulting in a maximum jitter of $J_2 = d_{M2} - d_m$, in which d_{M2} is the new maximum delay whereas d_m is the unchanged minimum delay through the network due to propagation and minimum processing delays. The resulting $R_D(t)$ function through the corresponding time period, is bounded by a minimum and a maximum segment denoted by R_{Dm} and R_{DM} respectively (see Figure 3). The segments are given below:

$$R_{Dm}(t) = \begin{cases} \rho_1(t - h) + v, & h \leq t \leq t_{s_m} \\ \rho_2(t - t_{s_m}) + A, & t > t_{s_m}, \end{cases} \quad (9)$$

$$R_{DM}(t) = \begin{cases} \rho_1(t - (h + J_1)) + v, & h + J_1 \leq t < t_{e_M} \\ A, & t_{e_M} \leq t \leq t_{s_M} \\ \rho_2(t - t_{s_M}) + A, & t > t_{s_M}, \end{cases} \quad (10)$$

where $J_1 = d_{M1} - d_m$, and d_{M1} is the worst case delay when the sending rate is ρ_1 . As shown in Figure 3, $t_{s_m} = t_{e_m}$, $t_{e_M} = t_{e_m} + J_1$, and $t_{s_M} = t_{s_m} + J_2$.

3.1.2 The Case of Rate Increase

In the case that the sending rate ρ_1 is increased to the new rate ρ_2 , the corresponding maximum delay becomes smaller after the decrease. The resulting $R_D(t)$ function through the corresponding time period, is again bounded by a minimum and a maximum segment denoted by R_{Dm} and R_{DM} respectively. The segments are given below:

$$R_{Dm}(t) = \begin{cases} \rho_1(t - h) + v, & h \leq t \leq t_{s_m} \\ \rho_2(t - t_{s_m}) + A, & t > t_{s_m}, \end{cases} \quad (11)$$

$$R_{DM}(t) = \begin{cases} \rho_1(t - (h + J_1)) + v, & h + J_1 \leq t < t_{e_M} \\ r(t - t_{e_M}) + A, & t_{e_M} \leq t \leq t_{e_r} \\ \rho_2(t - t_{s_M}) + A, & t > t_{e_r}, \end{cases} \quad (12)$$

where $J_1 = d_{M1} - d_m$, with d_{M1} is the worst case delay when the sending rate is ρ_1 , and r is the link bandwidth. Referring to Figure 4, $t_{s_m} = t_{e_m}$, $t_{e_M} = t_{s_m} + J_1$, $t_{s_M} = t_{s_m} + J_2$ and $t_{e_r} = \frac{rt_{e_M} - \rho_2 t_{s_M}}{r - \rho_2}$.

3.2 Smoothing Algorithms for Variable Network Jitter

The major approaches identified in the literature [1, 6, 8, 10] to find an optimally-smoothed transmission schedule use ideas from the following theories:

1. Theory of majorization.
2. Dynamic Programming.
3. Ad-hoc methods.

In our example optimal algorithm, we show how to modify the dynamic programming based algorithm proposed by Jiang et al. [6]. Other algorithms found in the literature [1, 8, 10] can be modified similarly to optimize their transmission schedule based on the variable network jitter.

The goal of the smoothing algorithm is to derive the optimal schedule for the function $R_D(t)$ only, since $R_S(t)$ can be made equal to $R_{Dm}(t - d_m)$. We are going to systematically compute the functions $R_{Dm}(t)$ and $R_{DM}(t)$ for every time point we apply the algorithm, and keep only the segments that minimize our cost function at every step. By quantizing everything to the frame boundaries, the computation cost of the algorithm becomes smaller.

Let us denote by $S_k(b, \rho_r, \rho_t)$ the state of our system at frame k , with buffer occupancy of b bits, reserved transmission rate of ρ_r , and actual transmission rate of ρ_t . We form a set of eligible states for the next step ($k + 1$) that can be reached from the current state without violating the corresponding conditions for buffer under/overflow. We call this set of states by $E(b, \rho_r, \rho_t)$. Therefore,

$$E_k(b, \rho_r, \rho_t) = \{S_{k+1}(b', \rho'_r, \rho'_t) \mid 0 \leq b' \leq B_D, \rho'_r \leq r, \rho'_t \leq \rho'_r\}.$$

It is evident that if no such states exist, then $E_k(b, \rho_r, \rho_t)$ is an empty set. At each step of the algorithm we find the state that results in the minimum cost. Therefore, we form a shortest path through the states that corresponds to the optimal schedule for the given cost function. The cost function is assumed to be additive. The dynamic programming algorithm is given in Figure 6.

We can work in a similar way to design the corresponding algorithm that computes the optimal transmission schedule of an MPEG-2 Transport Stream over a packet-switched network of variable jitter.

We continue our discussion with the presentation of the impact of data present in the stream that do not enter the decoder buffer on the computation of the transmission schedule.

4 Sender-based Smoothing

The video smoothing algorithms presented in the literature solve the problem of smoothing a video stream as seen on the receiver side, i.e., how the data of the video stream should arrive at the receiver to minimize a specific cost function. However, this does not correspond necessarily to the way the video stream should be sent from the sender to the receiver. The problem arises from the fact that a video stream often contains data which may not be relevant to the data decoded at the receiver side. This data needs not be inserted in the decoder buffer and therefore it should not be considered in any computation involving the decoder buffer. Due to this, transmission schedules computed by existing smoothing algorithms cannot guarantee that the receiver buffer will not under/overflow. We now proceed with the presentation of the problem and possible solutions for the case of elementary video streams. The case of MPEG-2 Transport Streams is discussed but due to space constraints is not presented. The interested reader is referred to [14] for more information.

4.1 The Elementary Video Stream Case

The contents of an elementary video stream often include data that are either not used by the decoder, or not inserted into the decoder buffer, or replacing other data previously transmitted (e.g. quantization tables). More specifically, in the case of MPEG-encoded elementary video streams, the stream contains among others the Group-of-Pictures (GOP) headers, the frame headers, and the optional quantization vector tables. Although all the data of a video stream are sent from the sender to the receiver, not all are placed in the decoder buffer. An example MPEG elementary video stream is shown in Figure 7.

```

1. k = -1; /* Initialize */
2.  $S_{k+1}(b_o, \rho_{r_o}, \rho_{t_o}) \leftarrow S_0(0, 0, 0)$ ;
3. For every step  $k \geq 0$ 
4.   Set  $S_k(b_o, \rho_{r_o}, \rho_{t_o})$  as the initial state;
5.    $E_k(b_o, \rho_{r_o}, \rho_{t_o}) \leftarrow \emptyset$ ;
6.   For every  $\rho_r \leq r$ 
7.     For every  $\rho_t \leq \rho_r$ 
8.       Compute new jitter  $J \leftarrow f(\rho_r)$ ;
9.       Compute  $R_{Dm}$  and  $R_{DM}$  for  $t \in [d_m + \frac{k}{f}, d_m + \frac{k+1}{f}]$ ;
10.      Check feasibility conditions;
11.      if feasibility conditions hold
12.        Add  $S_{k+1}(b', \rho_r, \rho_t)$  to set  $E_k(b_o, \rho_{r_o}, \rho_{t_o})$ ,
           where  $b' = \lceil R_{Dm}(d_m + \frac{k+1}{f}) - R_N(d_m + \frac{k+1}{f}) \rceil$ ;
13.        Compute corresponding cost  $C_{S_k(b_o, \rho_{r_o}, \rho_{t_o}) \rightarrow S_{k+1}(b', \rho_r, \rho_t)}$ ;
14.      Endif;
15.    Endfor;
16.  Endfor;
17.   $C_m = \infty$ ; /* initialize current temporary cost */
18.  For every state  $S_{k+1}(b', \rho_r', \rho_t') \in E_k(b_o, \rho_{r_o}, \rho_{t_o})$ 
19.    if  $C_{S_k(b_o, \rho_{r_o}, \rho_{t_o}) \rightarrow S_{k+1}(b', \rho_r', \rho_t')} < C_m$ 
20.       $S_{k+1}(b_o, \rho_{r_o}, \rho_{t_o}) = S_{k+1}(b', \rho_r', \rho_t')$ ;
21.       $C_m = C_{S_k(b_o, \rho_{r_o}, \rho_{t_o}) \rightarrow S_{k+1}(b', \rho_r', \rho_t')}$ ;
22.    Endif;
23.  Endfor;
24.  Total Minimum Cost up to slot  $(k + 1) = C_{S_0(0,0,0) \rightarrow S_{k+1}(b_o, \rho_{r_o}, \rho_{t_o})}$ ;
25. Endfor;

```

Figure 6: Dynamic programming algorithm to compute the optimal transmission schedule of an elementary video stream while considering networks with variable jitter dependent on the reserved rate.

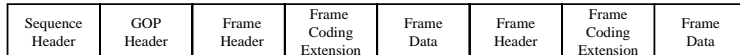


Figure 7: Various fields (headers and data) of an MPEG elementary video stream.

Two methods are proposed to derive a transmission schedule for an elementary video stream that takes into account the non-relevant data found in the stream: the *rate adaptation* method which follows a bottom-up approach, and the *forward schedule computation* method which follows a top-down approach.

Rate Adaptation

In the first method, we start by finding the schedule (i.e., $R_D(t)$) for the relevant data on the receiver side using any of the smoothing algorithms found in the literature. Assuming that the transmission rate changes at the points at which the resulting $R_D(t)$ function changes its slope, we need to compute the corresponding transmission rate for each segment of the computed schedule. Obviously, the corresponding transmission rate is always going to be greater than the rate of each segment of the original function $R_D(t)$.

To find the corresponding transmission rate for a specific segment of $R_D(t)$ we need to compute the amount of data that is transmitted during its corresponding period T (see Figure 8). Then, we need to find the beginning and the end of this data segment in the elementary video stream. We will refer to

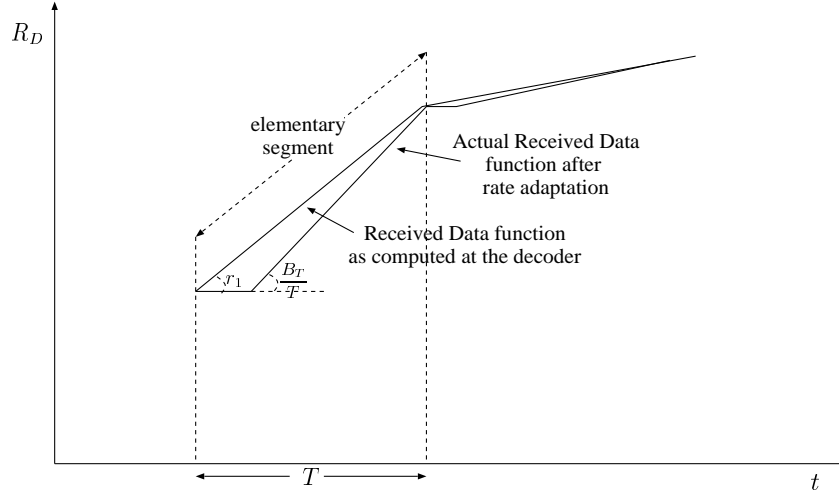


Figure 8: Discrepancy of a rate segment computed using an optimal algorithm and the resulting segment after the transport rate adaptation procedure.

the resulting segment as *elementary segment*. The goal is to change the transmission rate through the elementary segment in such a way that the average rate of the corresponding data segment of the decoded program remains constant and equal to the one already computed in the function $R_D(t)$. We refer to this procedure as *rate adaptation*. Assuming that the elementary segment contains B_T amount of data, then the transmission rate after the rate adaptation method becomes B_T/T .

In the general case, rate adaptation may not be feasible if consecutive relevant data from the video stream are followed by other data for long time intervals. However, in the case of elementary video streams, we can assume that all the non-relevant to the decoded stream data reside in the beginning of each frame and their size is small compared to the frame sizes. Therefore, the transmission rate assigned to an elementary segment will result in a receiver arrival function of the relevant data that follows closely the function $R_D(t)$ computed in the first step.

The procedure described above can be done on a per-segment basis as well. In such a case, after the computation of a segment of the function $R_D(t)$, the corresponding sender transmission rate should be computed and then the resulting arrival function on the receiver side should be checked for feasibility again. If the resulting arrival function is not feasible, the segment of $R_D(t)$ is recomputed and the procedure repeats itself.

Forward Schedule Computation

A more straightforward approach for the computation of a smoothed transmission schedule for an elementary video stream is to optimize for the entire video stream from the beginning while checking whether the feasibility conditions for the relevant data hold on the receiver side. More specifically, we select a transmission rate for every data segment that contains the non-relevant data and the frame data for a certain frame, and then check the feasibility conditions on the receiver side for the resulting relevant data (frame data). Under the assumption that the non-relevant data are a prefix of the frame data for each frame, and the transmission rate remains constant over the duration of a frame (corresponding to a specific frame rate) of the video stream, this procedure becomes the same as the rate adaptation method described previously when the latter method is reversed. In the general case, however, in which the points at which the transmission rate is allowed to change is not quantized at the frame boundaries, this method will result in a feasible schedule for the elementary video stream other than the one generated by the rate adaptation method. Dynamic programming can be also applied in this case to generate an optimal schedule for the transmission of the stream.

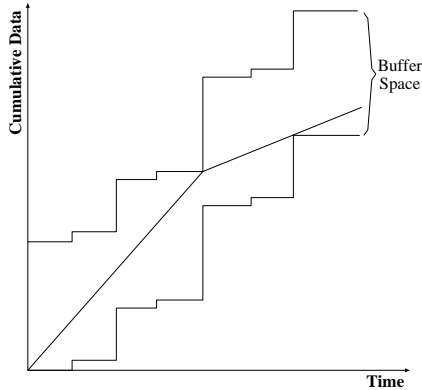


Figure 9: Example of a typical smoothed schedule of a video stream.

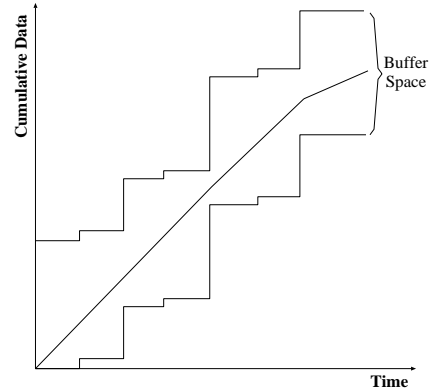


Figure 10: Example of a clock-aware smoothed schedule of a video stream.

4.2 The MPEG-2 Transport Stream Case

The MPEG-2 Transport stream format multiplexes one or more programs into a single stream. Since only one program is typically decoded by a specific receiver, all the remaining data of the stream are thrown away on the receiver side. Even in the case of a *Single-Program Transport Stream (SPTS)*, most of the data residing in the headers of the transport packets and the *packetized elementary stream (PES)* packets are used for data integrity, clock synchronization and other purposes and, therefore, are not inserted into the system decoder buffer. Similarly to the elementary video stream case, the two methods described previously can be applied here as well. The interested reader is referred to [14] for elaborate information on the proposed solutions.

We now proceed with a discussion of possible effects of the transmission schedule on the clock recovery process at the decoder end in the case that the decoder clock is derived from the incoming stream.

5 Clock Recovery based Smoothing

The delivery of a video stream through a packet-switched network is usually associated with a specific service provisioning which corresponds to certain categories of applications. The most common case includes the regular applications that do not require the receiver to reconstruct a clock from the incoming video stream to be used by the decoding process (e.g. typical video streaming applications carried over the Internet). In these applications, the decoder may operate from a local free-running clock which is typically used to forward the decoded video frames to the actual screen according to the frame rate of the video stream.

Another category of applications requires a clock to be reconstructed from the incoming video stream. The reconstructed clock can be used in different functions such as for estimating the packet delay through the network, or synthesizing a chroma sub-carrier for the composite video signal of the TV set in the case of broadcasting applications. The use of applications with stringent clock specifications requires a careful design of not only the decoder but also the video stream's delivery procedure.

The transmission schedules for a given video stream as computed by the smoothing algorithms found in the literature [1, 6, 8, 10] maximize the time intervals at which a transmission rate is used without causing under/overflow of the receiver buffer. An example of a short segment of a transmission schedule is given in Figure 9. It is evident that the receiver buffer is forced to almost underflow after almost overflowing and vice versa. Hence, any estimation of the average rate of the stream, or any reconstruction of a stable clock from the incoming video stream becomes a non-trivial task.

In applications in which the incoming video stream is used for clock reconstruction at the receiver, a smoothing algorithm should compute the transmission schedule in such a way to facilitate the clock recovery process. A typical solution to this problem is to maintain a fixed buffer occupancy in the receiver buffer (e.g. maintain the receiver buffer in half-full status). An example *clock-aware* smoothed schedule is shown in Figure 10 in which the schedule tries to maintain constant buffer occupancy in the receiver buffer.

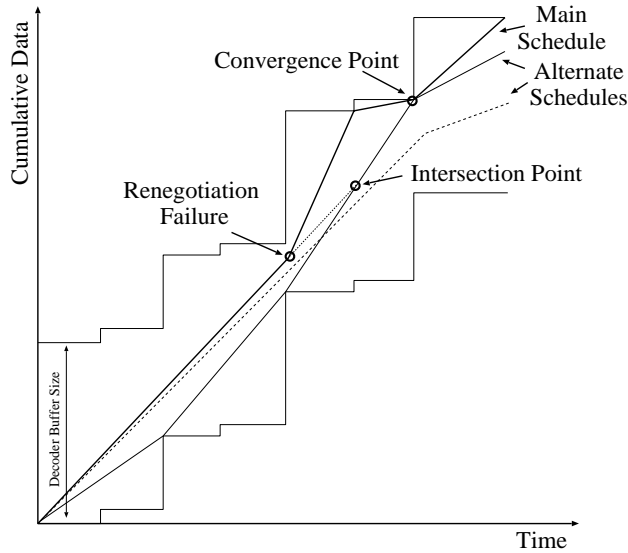


Figure 11: Example of alternate schedules computed for a video stream.

We will now discuss possible ways that the actual transmission schedule remains feasible even in the case that a specific renegotiation request of the precomputed smoothed schedule cannot be granted.

6 Renegotiation Decisions

The transmission schedule of a video stream computed by a video smoothing algorithm contains the sequence of transmission rates that guarantees no under/overflow of the receiver buffer. The sender negotiates the transmission rate according to the schedule with the network which in turn provides the necessary resources. However, a request to increase the transmission rate of the sender may be rejected by the network due to insufficient resources. The service is statistical and only guarantees that the probability of such a rejection is relatively low.

On a renegotiation failure, the sender can use different techniques to guarantee that the receiver will not under/overflow. The first technique utilizes dynamic requantization of the video stream so that the resulting stream can be sent with lower transmission rate [3].

In the second technique [5], if the video stream is encoded using a scalable encoding process in which a base layer is combined with a set of enhancement layers, the sender decides to drop specific enhancement layers to lower the actual network resource demand and, therefore, the transmission rate of the stream.

In our proposed technique, the sender precomputes a set of transmission schedules which have points of convergence. In the case that a specific renegotiation request is not granted for a specific transmission schedule, the sender continues sending at the current rate while trying to renegotiate at some later time instant. The new renegotiation point can be selected as the intersection point between the altered schedule and some other schedule of the set as shown in Figure 11. The renegotiated rate should then be the transmission rate corresponding to the second transmission schedule at the intersection point. The old schedule can be retraced at the next convergence point between the two schedules.

7 Conclusions

We addressed the problem of transporting stored multimedia traffic over a packet-switched network using smoothing algorithms to increase the network utilization and decrease the bandwidth variability of the traffic stream. We elaborated on the smoothing procedure and its end-to-end effects by identifying and proposing solutions for several critical points that should be taken into account by any smoothing algorithm to generate feasible transmission schedules. The general points identified along with the proposed solutions included: (i)

the smoothing procedure under possible variable jitter throughout the network, (ii) the consideration of the unused data within the stream when the transmission schedule is computed, (iii) the clock recovery problem, and (iv) the use of alternate schedules on a renegotiation failure.

A number of issues remains open. The adaptation of a precomputed transmission schedule for a stored stream and specific network parameters for transmission under different network parameters without recomputing the whole schedule needs further investigation. Also, the impact of the transmission schedule on the clock recovery process at the decoder for applications with stringent clock requirements such as broadcast TV, should be examined in more detail. Finally, a thorough understanding of how a set of smoothed video streams can be combined and sent over a packet-switched network to increase its utilization needs to be studied.

References

- [1] W. Feng. *Buffering Techniques for Delivery of Compressed Video in Video-on-Demand Systems*. Kluwer Academic Publishers, 1997.
- [2] S. J. Golestani. A self-clocked fair queueing scheme for broadband applications. In *Proceedings of IEEE INFOCOM '94*, volume 2, pages 643–646, June 1994.
- [3] M. Grossglauser, S. Keshav, and D. Tse. RCBR: a simple and efficient service for multiple time-scale traffic. In *Proceedings of ACM SIGCOMM '95*, volume 25, pages 219–230, Cambridge, MA, USA, August 1995.
- [4] International Organization for Standardization. *Information Technology — Generic Coding of Moving Pictures and Associated Audio: Systems, Recommendation H.222.0, ISO/IEC 13818-1*, first edition, April 1996.
- [5] International Organization for Standardization. *Information Technology – Generic Coding of Moving Pictures and Associated Audio Information: Video, Recommendation ITU-T H.262, ISO/IEC 13818-2*, first edition, May 1996.
- [6] Z. Jiang and L. Kleinrock. A general optimal smoothing algorithm. In *Proceedings of IEEE INFOCOM '98*, volume 1, 1998.
- [7] E. W. Knightly. H-BIND: A new approach to providing statistical performance guarantees to VBR traffic. In *Proceedings of IEEE INFOCOM '96*, volume 3, pages 1091–1099, San Francisco, CA, March 1996.
- [8] J. M. McManus and K. W. Ross. Video-on-demand over ATM: Constant-rate transmission and transport. *IEEE Journal on Selected Areas in Communications*, 14:1087–1098, August 1996.
- [9] A. K. Parekh and R. G. Gallager. A generalized processor sharing approach to flow control in integrated services networks: The multiple-node case. *IEEE/ACM Transactions on Networking*, 2(2):137–150, April 1994.
- [10] 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. In *Proceedings of ACM SIGMETRICS '96*, volume 1, 1996.
- [11] D. Stiliadis and A. Varma. A general methodology for designing efficient traffic scheduling and shaping algorithms. In *Proceedings of IEEE INFOCOM '97*, Kobe, Japan, April 1997.
- [12] D. Stiliadis and A. Varma. Efficient fair-queueing algorithms for packet-switched networks. *IEEE/ACM Transactions on Networking*, pages 175–185, April 1998.
- [13] C. Tryfonas. *Video Transport over Packet-Switched Networks*. PhD thesis, University of California at Santa Cruz, 1999.
- [14] C. Tryfonas. Practical considerations for smoothing multimedia traffic transported over packet-switched networks. Technical Report 2001-12-5, Sprint Advanced Technology Laboratories, December 2001. Available at <http://www.sprintlabs.com/People/tryfonas/research.htm>.
- [15] L. Zhang. VirtualClock : A new traffic control algorithm for packet-switched networks. *ACM Transactions on Computer Systems*, 9(2):101–124, May 1991.