An Algorithm for Dynamic Bandwidth Allocation of MPEG Videos*

Errin W. Fulp† and Douglas S. Reeves‡
Departments of ECE and CSC
North Carolina State University

1 Introduction

Multimedia traffic (voice and video) in networks requires control of such quality of service (QoS) parameters as average delay, delay variation, and maximum loss rate. To ensure that QoS requirements will be met, the normal approach is to require the application to describe its traffic, using such statistics as the period, deadline, average and worst-case processing time or packet size, etc. These statistics are often conservative estimates, and are statically declared. Based upon these statistics, a calculation is made of the resources (buffer space, processing time-slice, network and disk bandwidth, etc.) needed to provide the requested QoS.

QoS guarantees for multimedia traffic are difficult for three reasons. First, when the traffic is processed in real-time (for interactive applications), it is impossible to predict precisely what the actual behavior will be. Second, the traffic itself may be highly variable, which makes a peak-rate model excessively conservative. Third, compressed video can exhibit long-range dependency [4][6], which implies that the standard statistical traffic models are probably inadequate. For these reasons efficient resource allocation, along with good control of QoS, is a challenging problem.

To date a variety of bandwidth allocation methods have been proposed. *Off-line methods* will make allocation decisions before any traffic is transmitted, and may assume that the traffic is available for analysis (such as in the case of stored video) before transmission. An off-line method may allocate one (static) resource level for the duration of the application, or may renegotiate the resource level at various times over this duration. An example of a static off-line method is the peak-rate allocation which is used for most real-time applications. This approach has the advantages of simplicity and predictability, but suffers from the problems noted above. Off-line, renegotiation methods can result in significantly better resource utilization [2], but require complete information about the source traffic. For video, this means access to the entire video's traffic trace, which is not possible for interactive applications.

On-line methods periodically renegotiate the resource allocation based upon a prediction of the future traffic behavior. This prediction is derived from measurements of the traffic that have been observed so far, and the QoS that has been experienced. Such methods, including [1] [3] [7] [8] do not have the problems associated with off-line methods. However, to date no on-line method has had the ability to tightly control QoS for such difficult applications as transmission of compressed video. In addition, most methods suffer from a large number of renegotiations, and/or relying on a very complex measurement and allocation algorithm.

In this paper we present an on-line renegotiation method called the Dynamic Search Algorithm (DSA+) [5]. We demonstrate the use of DSA+ to control the loss rate of actual MPEG VBR videos by the appropriate allocation of bandwidth. We investigated the bandwidth savings of DSA+ and the impact of parameter selection when controlling various MPEG videos. Experiments show substantial savings over the optimal off-line static (peak-rate) allocation, and the robustness of DSA+ with respect to the initial parameter settings.

*This work was supported by AFOSR grant F49620-96-1-0061. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the AFOSR or the U.S. Government.

†ewfulp@eos.ncsu.edu
‡reeves@eos.ncsu.edu

curr_error $\leftarrow \ln(P_n/Q_l)$
prev_error $\leftarrow \ln(P_{n-1}/Q_l)$
if$((P_{0\ldots n} > Q_l)$ **AND** $(P_n \leq Q_l))$**then**
    $U_{n+1} \leftarrow 2 \times U_n$
else
    if (curr_error $\times$ prev_error ≤ 0)**then**
        $U_{n+1} \leftarrow 2 \times U_n$
    endif
    if$(P_n > Q_l)$**then**
        $\mu_{n+1} \leftarrow \mu_n + K \times \ln(P_n/Q_l)$
    else
        $\mu_{n+1} \leftarrow \mu_n + \frac{K}{2} \times \ln(P_n/Q_l)$
    endif
endif

Figure 1: DSA+ algorithm

2 Algorithm Description

2.1 System Model

DSA+ dynamically renegotiates the server rate (bandwidth) of a finite capacity queue in order to meet a desired QoS. A cell is the fixed-length unit of traffic storage and transmission. In this paper the QoS of interest is the cell loss probability (CLP) of a single source. Cell arrivals to and losses from this queue are monitored throughout the duration of the application. Rate changes are renegotiated at discrete instances of time. We denote the nth renegotiation instant as t_n, and the interval between renegotiation points t_n and t_{n+1} as the nth update interval, U_n. The service rate during U_n is constant (with the exception noted below), and is denoted μ_n.

During the nth interval, let the number of arrivals be represented by A_n and the number of losses as L_n. The CLP of the nth interval is then calculated as $P_n = L_n/A_n$. The cumulative CLP of all the intervals up to and including the nth is

$$P_{0\ldots n} = \sum_{i=0}^{n} \frac{L_i}{A_i}$$

The CLP desired by the user is denoted Q_l.

The goal of DSA+ is to adjust the server rate to a minimum value μ^*, while the CLP approaches Q_l. Secondary goals are simplicity of implementation, robustness, and minimizing the number of renegotiations.

2.2 Algorithm for Dynamic Resource Allocation

At each renegotiation point DSA+ adjusts the server rate according to the following formula:

$$\mu_{n+1} \leftarrow \mu_n + \frac{K}{\alpha} \times \ln\left(\frac{P_n}{Q_l}\right), \alpha = \begin{cases} 1 & if \quad P_n > Q_l \\ 2 & if \quad P_n \leq Q_l \end{cases} \tag{1}$$

In this simple closed-loop control algorithm, the server rate is increased proportionally to the difference in the desired and measured QoS. The use of the logarithm of the ratio of P_n to Q_l is appropriate due to the very small loss rates that are normally desired. K determines how much the server rate can be increased or decreased at each instant. Parameter α allows the rate to be increased twice as fast as it can be decreased.

Figure 1 shows the complete algorithm at one renegotiation instant t_n. As seen in the figure, the renegotiation interval is lengthened (doubled) in two cases: if the cumulative CLP and the CLP during the most recent interval (P_n) are both better than required, or when P_n and P_{n-1} are on different sides of (one greater than, the other less than) the desired CLP.

A potential problem with the algorithm as shown is that the traffic characteristics may change drastically during a renegotiation interval, while the server rate cannot be renegotiated. This can lead to excessive QoS

violations. To reduce the severity of this problem, we introduce the use of interrupts. At fixed-length sub-intervals, an interrupt is generated if both P_n and $P_{0..n}$ are greater than the desired loss rate Q_l. In this case, the server rate is increased immediately according to equation 1, rather than waiting until the end of the renegotiation interval. The renegotiation interval itself, however, is not changed by an interrupt. Interrupts make DSA+ more responsive to sudden, severe traffic changes, which should occur infrequently (as we show below).

Initially the user must assign the following values: desired CLP Q_l, initial renegotiation interval U_0, interrupt sub-interval I, K and the initial server rate μ_0. U_0, I, and K may depend on the source traffic, but their selection primarily impacts the number of renegotiations and the efficiency of the allocation. Initial variable selection is addressed in the following sections.

3 Numerical Results

In this section the performance of DSA+ is investigated using fifteen MPEG-compressed videos. All traces were obtained from Oliver Rose at the University of Würzburg, Germany [9][1]. Each trace is a thirty minute segment of the original video and were encoded using the same MPEG-1 encoder card. Relevant statistics of each video are presented in [5] and [9]. As reported in [9], the Hurst parameters indicate all videos exhibit long-range dependency, and significant peak-to-mean ratios. Therefore it is evident that these are very difficult sources to regulate, and to date there has been no successful attempt to efficiently manage them on-line.

For each experiment the system described in section 2.1 was simulated. The desired QoS was a CLP of 1×10^{-3} and the queue capacity was 80 ATM cells (48 byte payload) [8]. For each I, B or P MPEG frame, the equivalent number of ATM cells was determined. The cell arrival times were then uniformly distributed over the duration of the frame. This process was repeated for each frame until the end of the trace was reached. No smoothing, multiplexing, filtering or quantization changes of any kind were made to the videos. We consider these experiments to be a "worst-case" test of DSA+.

Table 1 shows the performance of DSA+, and an off-line peak rate allocation algorithm. In a sense, this comparison is unfair to DSA+, since we are comparing an on-line algorithm to an off-line one. An on-line peak rate algorithm would have to overestimate the traffic by a substantial percentage to be cautious. Another difference is that peak rate allocation would result in 0 losses, while DSA+ was targeted for a loss rate of 10^{-3}. Small but non-zero losses are considered to be acceptable for typical multimedia applications. Rather than being a weakness, we consider the ability to manage different QoS targets, based upon the user's needs, to be a strength of any on-line algorithm.

For DSA+, the initial renegotiation interval was 4 seconds, the interrupt interval 0.5 seconds and K was 100 Kbps. The primary metric for comparison is the number of bits used to transmit the video, or equivalently, the area under the allocation curve for the duration of the video. As indicated in table 1, DSA+ requires fewer bits than peak rate allocation. Savings of DSA+ as compared to peak allocation ranged from 12.7% to 57.3%. In each experiment DSA+ reduced the bandwidth allocated while maintaining the desired QoS. Allocation changes ranged from 17 to 53 for the individual videos, which is very low as compared to other on-line algorithms [5].

3.1 Sensitivity to Initial Parameter Values

As described above, three initial parameters must be specified in order to use DSA+: the first renegotiation interval (U_0), the rate adjustment coefficient K, and the interrupt sub-interval I. A capable dynamic allocation method should be relative insensitive to the initial parameter values, so that user insight into these parameters is not a requirement. We ran three experiments to investigate whether this was the case. For each experiment, we set the parameters to a particular value and simulated the behavior of DSA+, for all 15 videos. Then, the average number of bits and number of renegotiations per video was calculated and plotted.

In the first experiment, K was varied while U_0 and I were fixed at 4 and 0.5 seconds respectively. Figure 2 shows the results. The number of renegotiations dropped by a relatively small amount (10%) as K varied

[1] Traces can be obtained from the ftp site ftp-info3.informatik.uni-wuerzburg.de in the directory /pub/MPEG

Video		DSA+				Peak
	Peak/ Mean	Number of Reg. Updates	Number of Interrupts	Bits Used ($\times 10^9$)	CLP ($\times 10^{-3}$)	Bits Used ($\times 10^9$)
Asterix (cartoon)	6.59	22	13	3.69	0.6	5.9
ATP Tennis	8.72	21	8	5.6	0.938	7.63
Formula 1 Race	6.58	20	6	5.69	0.395	8.10
Goldfinger	10.1	26	19	5.19	0.874	9.79
Jurassic Park	9.15	23	12	2.89	0.967	4.78
Movie Review	12.1	20	10	4.79	0.580	6.9
Mr. Bean	13.0	30	22	3.92	1.0	9.17
MTV	9.31	32	21	6.38	0.911	9.17
News (German)	9.01	20	9	4.72	0.31	7.78
Silence of the Lambs	18.4	30	16	3.09	1.0	5.37
Simpsons (cartoon)	12.9	32	9	4.32	0.793	9.62
Soccer (World Cup)	6.9	15	10	6.53	1.0	7.48
Super Bowl (1995)	5.99	22	12	4.05	0.344	5.63
Talk (German)	7.34	14	3	2.41	0.512	4.27
Terminator	7.29	28	3	1.87	0.438	3.18

Table 1: Algorithm performance.

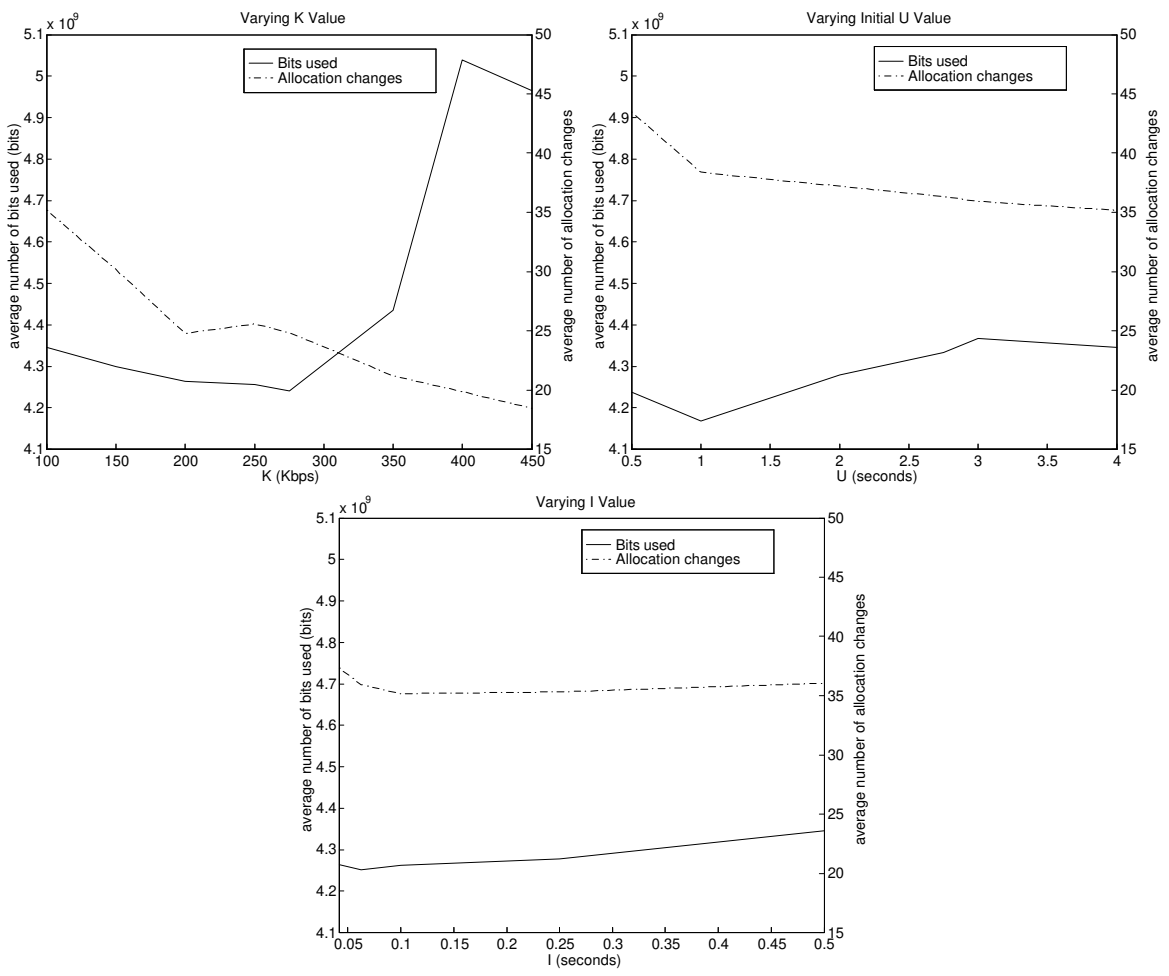

Figure 2: Varying initial parameters K, U_0 and I independently.

from 50 to 450 Kbps. The number of bits, however, increases significantly as K exceeds 350 Kbps, indicating some sensitivity to this parameter value. Experiment 2 varied U_0 as K was fixed at 100 Kbps and I was fixed at 0.5 seconds. Experiment 3 varied I as K was fixed at 100 Kbps and U_0 was fixed at 4 seconds. The results of both experiments are shown in Figure 2. In both cases, the initial parameter value has almost no effect on the number of bits needed, or the number of renegotiations. In general, for MPEG videos smaller K values (100-350 Kbps) are better, since the additional number of renegotiations is not significant compared to the savings in bandwidth.

4 Conclusions

This paper presented an on-line algorithm, DSA+, which efficiently allocates resources to provide a required QoS. DSA+ was used to manage the bandwidth of MPEG traces with a specified allowable cell loss probability. Fifteen actual MPEG traces were collected and used in all the experiments to measure the performance of the algorithm. We were interested in minimizing both the bandwidth allocated and the number of renegotiations. As compared to an off-line peak-rate allocation, DSA+ saved 13–58% in bandwidth. The number of renegotiations for each video was between 17 and 53, which seems acceptably low. Our experiments also indicate the algorithm is fairly insensitive to the choice of initial parameter values. More details about our work on dynamic resource allocation can be found at the web site `ftp://ftp.csc.ncsu.edu/pub/rtcomm/rtcomm.html`

While the focus of this paper was the bandwidth allocation of MPEG videos, DSA+ may be useful for other real-time applications. Examples include CPU scheduling and disk bandwidth management. In both cases the central idea is to provide guaranteed service to variable traffic, with the minimum amount of resources and user input.

References

[1] A. Adas. Supporting Real Time VBR Video Using Dynamic Reservation Based on Linear Prediction. In *IEEE INFOCOM'96*, pages 1467–1483, 1996.

[2] K. Chang and H. T. Kung. Efficient Time-Domain Bandwidth Allocation in A Video-on-Demand System. In *The Fifth ICCCN-International Conference On Computer Communications and Networks*, pages 172–178, Oct. 1996.

[3] S. Chong, S.-Q. Li, and J. Ghosh. Predictive Dynamic Bandwidth Allocation for Efficient Transport of Real-Time VBR Video over ATM. *IEEE J. on Selected Areas in Communication*, 13(1):12–23, January 1995.

[4] A. Erramilli, O. Narayan, and W. Willinger. Experimental Queueing Analysis with Long-Range Dependent Packet Traffic. *IEEE/ACM Transactions on Networking*, 4(2):209–223, Apr. 1996.

[5] E. W. Fulp, D. S. Reeves, and Y. Viniotis. Dynamic Bandwidth Allocation for VBR Sources. Technical Report TR-96/45, North Carolina State University Department of Electrical and Computer Engineering, Oct. 1996.

[6] M. W. Garret and W. Willinger. Analysis, Modeling and Generation of Self-Similar VBR Video Traffic. In *SIGCOMM'94*, pages 269–280, London, 1994.

[7] I. Hsu and J. Walrand. Dynamic Bandwidth Allocation for ATM Switches. To appear in the *Journal of Applied Probability*, Sept. 1996.

[8] S. Rampal, D. Reeves, and I. Viniotis. Dynamic Resource Allocation Based on Measured QoS. In *The Fifth ICCCN-International Conference On Computer Communications and Networks*, pages 24–27, 1996.

[9] O. Rose. Statistical Properties of MPEG Video Traffic and Their Impact on Traffic Modeling in ATM Systems. Technical Report 101, University of Würzburg Institute of Computer Science, Feb. 1995.