On-line Dynamic Bandwidth Allocation*

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

Abstract

Network multimedia applications require certain performance guarantees that can be provided through proper resource allocation. Allocation techniques are needed to provide these guarantees as efficiently as possible since resources are limited. This paper presents an allocation method called Dynamic Search Algorithm (DSA+). DSA+ is an on-line algorithm that dynamically adjusts the resource allocation based upon the measured quality of service. Advantages of DSA+ include efficient use of resources, reasonable implementation cost and stringent quality of service control. In this paper we demonstrate how DSA+ dynamically allocates bandwidth to achieve a given loss rate for actual variable bit rate MPEG videos. Performance and cost advantages over other allocation methods are presented, as well as allocation for multiple hop connections.

1 Introduction

As the use of multimedia applications increases, so are the demands for resources required to support them. Network resources such as bandwidth of each physical link, buffer space and processing time at each node, should be allocated in a cost-effective manner. Each application expects the network to provide a desired quality of service (QoS). QoS measurements include bounds on the cell loss probability, cell delay, etc. The service provider is interested in providing the desired QoS, but as efficiently as possible. For these reasons, allocation methods are needed to allocate resources and to provide QoS guarantees.

Conventional approaches to resource allocation rely on predetermined traffic characteristics. The amount of resources required to provide the QoS is calculated using these values. These techniques experience the following fundamental problems. First, the source characteristics may not be known ahead of time. In the case of live or interactive video, the user must guess at these characteristics. Second, parameters may not adequately characterize the source. It has been shown for MPEG-compressed video that long-range dependencies occur, which implies that standard statistical models are probably inadequate [7]. Third, the number of parameters required should be kept small, so to reduce the complexity of the allocation method. A layered n-space Markov Model may adequately characterize a source, nevertheless the computation of allocation amounts may become intractable for real time applications [3].

Current allocation methods can be categorized as either *off-line* or *on-line*. Off-line methods predetermine allocation amounts before transmission begins. Such a method may allocate one resource level (static) for the duration of the application, or may renegotiate the resource level at various times. An example off-line allocation is peak rate, which is used for most real time applications. This approach has several advantages including simplicity and predictability, but suffers from the problems noted above, as well as low resource utilization if the peak-to-mean ratio is high. Other off-line methods that renegotiate resource levels result in better utilization; on the other hand, they require complete control of the traffic source [5]. For example, the off-line method developed by Feng, et al. determines the minimum number of renegotiations (increases or decreases in resource allocation) required for the playback of a previously-stored MPEG video [5]. The video is then transmitted at the calculated rates, so to prevent buffer overflow and underflow. For interactive applications, where the traffic source is not known nor directly controllable, these methods are not suitable.

On-line methods periodically renegotiate re-

*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

source allocation based upon predicted traffic behavior [1] [4] [8] [10] [11] [13]. Predictions are derived from measurements of the traffic and/or QoS observations. Such methods do not have the problems associated with off-line methods and may be implemented by filters [1], neural networks [4], dynamic algorithms [8] [10] or some other sampling procedure [11] [13]. These methods also have the advantage of adjusting the resource allocation with respect to a desired QoS. To date no on-line method has the ability to tightly control the QoS for such difficult applications as transmission of compressed video. In addition, most methods suffer from a large number of renegotiations, and/or rely on a very complex measurement and allocation algorithm.

In this paper we present an on-line renegotiation method called the Dynamic Search Algorithm (DSA+) [6]. Dynamic algorithms (also referred to as adaptive algorithms) have been applied in various applications such as system identification, filtering and pattern recognition [2]. DSA+ encapsulates the general dynamic algorithm form with new features to better handle non-stationary sources. DSA+ allocates resources so to meet a desired QoS. In this paper DSA+ is used to control the loss rate of actual MPEG VBR videos by the appropriate allocation of bandwidth. Our results show that DSA+ is able to efficiently allocate resources with fewer renegotiations than other on-line or off-line algorithms. The algorithm also can efficiently allocate resources for multiple hop connections and still provide the required QoS.

The remainder of this paper is organized as follows. Section 2 describes the system model and DSA+ algorithm in detail. Section 3 presents the results of allocation experiments with VBR MPEG videos. A comparison with other allocation techniques is made as well as the application of DSA+ for multiple hop connections. Section 4 summarizes our findings and discusses issues for future work.

2 A new method

2.1 System model

Networks must allocate a variety of resources in order to provide the desired QoS for each application. In this paper we focus on the management of server rate to provide a desired cell loss probability (CLP).

DSA+ dynamically renegotiates the server rate (bandwidth) of a queue in order to meet a desired QoS. Cells, a fixed-length unit of traffic storage and transmission, arrive at a finite capacity queue and are serviced in a FIFO manner. Any cell arriving at a full queue is immediately lost. 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 and 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 and is denoted as 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

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

The CLP desired by the user is denoted Q_l. The goal of DSA+ is to adjust the server rate, so to provide the desired CLP, Q_l, as efficiently as possible with few renegotiations. Secondary goals are simplicity of implementation and robustness.

2.2 An algorithm for dynamic resource allocation

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

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

This dynamic algorithm updates the server rate, μ_n, based on the observed CLP during the nth interval. This measurement along with the desired CLP value, Q_l, are then used in the error function $\ln(P_n/Q_l)$. This non-linear error function provides either a positive or negative feedback value based on the observation taken during the most recent interval. Note the feedback becomes smaller as the measured CLP approaches the desired value, keeping the rate at a more stable value. The error function is also appropriate due to the very small loss rates that are normally desired. The constant K amplifies the response of the error function and this product ultimately determines how much the server rate can be increased or decreased. Parameter α allows the rate to be increased twice as fast as it

```
1   curr_error ← ln(P_n/Q_l)
2   prev_error ← ln(P_{n-1}/Q_l)
3   if((P_{0...n} > Q_l) AND (P_n ≤ Q_l))then
4       U_{n+1} ← 2 × U_n
5   else
6       if (curr_error × prev_error ≤ 0)then
7           U_{n+1} ← 2 × U_n
8       endif
9       if(P_n > Q_l)then
10          μ_{n+1} ← μ_n + K × ln(P_n/Q_l)
11      else
12          μ_{n+1} ← μ_n + (K/2) × ln(P_n/Q_l)
13      endif
14  endif
```

Figure 1: DSA+ algorithm. Numbers on the far left are for reference only.

can be decreased. This is done since, in an actual network, resources are more easily reduced than increased. Varying the gain is also beneficial if the source is non-stationary [2].

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 (lines 4 and 7). The interval is doubled on line 4 if the cumulative CLP ($P_{0...n}$) is worse than required and if the CLP during the most recent interval (P_n) is better than required. This will reduce the cumulative CLP towards the desired value, since the nth CLP is better than the desired CLP. The interval is doubled on line 7 when P_n and P_{n-1} are on different sides of (one greater than, the other less than) the desired CLP. Doubling the interval length also reduces the number of renegotiations required over time, a unique feature to both DSA+ and REQS [10].

As mentioned in the introduction, traffic sources such as MPEG-compressed video are complex due to their non-stationary behavior and long range dependencies. 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, called an interrupt interval I, an interrupt is generated if both P_n and $P_{0...n}$ are greater than the desired loss rate Q_l. The relationship between update and interrupt intervals can be seen in figure 2. In this

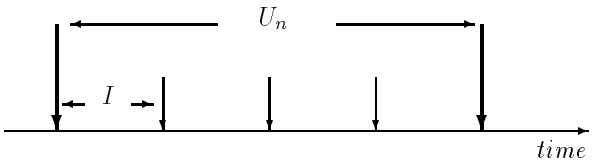

Figure 2: Time axis for updates and interrupts.

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. The use of interrupts allows DSA+ to be more responsive to sudden, severe traffic changes, which should occur infrequently. This is a unique and key element of DSA+.

Initially the user must assign the following values: initial renegotiation interval (U_0), interrupt subinterval (I), constant (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 later in this paper.

3 Numerical results

In this section the performance of DSA+ and other allocation techniques is investigated using actual MPEG-compressed traffic. 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) [10]. While the targeted QoS was cell loss probability, the queue size was selected to provide a maximum cell delay as well. The minimum allowed bandwidth was 1 Mbps. The queue length and the minimum bandwidth yields a maximum delay of 34 msec for any cell. These values were selected to provide a wide range of available bandwidths, however almost any combination of minimum bandwidth and queue size could have been chosen. A more restrictive bandwidth selection will only improve the performance of the algorithm.

Fifteen MPEG-compressed video traces were obtained from Oliver Rose at the University of Würzburg, Germany [12] [1]. Each trace is a thirty minute segment of the original video and each

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

μ_0 (Mbps)	K (Kbps)	U_0 (second)	I (second)
1000	100	4	0.5

Table 1: DSA+ initial parameter values.

was encoded with constant quality using the same MPEG-1 encoder card. Relevant statistics of each video are presented in [6] and [12]. As reported in [12], the Hurst parameters indicate all videos exhibit long-range dependency, and significant peak-to-mean ratios ranging from 18.4 to 4.63 based on average frames. 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 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 "hard-case" test of any on-line allocation technique.

We are interested in efficiently managing bandwidth, therefore two metrics are used. First, the number of renegotiations required for the entire simulation. This value is important since a large number of renegotiations will cause considerable strain on the signaling system of the network. Second, the number of bits reserved to transmit the video, or equivalently, the area under the allocation curve for the duration of the video. This measurement is important because we wish to transmit the video with as few bits as possible while maintaining the desired QoS. Minimizing the bits used can help increase the utilization of the network by providing more resources to other users.

3.1 Comparison with other methods

In this section DSA+ is compared to other allocation techniques: peak rate, Hsu's algorithm, and RED-VBR. This selection represents a variety of on-line and off-line allocation methods. Each is briefly introduced and explained. DSA+ initial parameters, the same for each video, are given in table 1. The selection of these parameters as well as the robustness of DSA+ to initial parameters is presented in [6].

Peak rate allocation was chosen since it is an accepted allocation method. To determine the exact peak rate requires the trace in advance. For that reason, this is an off-line method. In a sense this comparison is unfair to the remaining on-line methods. An on-line peak rate algorithm would require an overestimation of the traffic by a significant percentage to be cautious. Another difference is that peak rate allocation would result in zero losses, while other methods were targeted for a loss rate of 1×10^{-3}. Small but non-zero losses are considered to be acceptable for typical multimedia applications. Instead of a weakness, we consider the ability to manage QoS targets based upon the user's needs to be a strength of any on-line algorithm. Other off-line methods, such as Feng's algorithm [5], are not comparable, as they directly control the transmission of the source.

Hsu's method is a dynamic algorithm which has been proven to find the minimum bandwidth required for a stationary MMBP source [8]. This method was chosen since it is a simple on-line method that requires minimal source information. The algorithm renegotiates bandwidth over fixed length intervals, using previous loss measurements and a simple difference error function. Initial parameters for this algorithm were 4.5 Mbps for the initial server rate, 1 second for the interval and 1 Mbps for c the constant. No method of parameter selection was presented in the original paper. The values used were found to be the best from our experiments. It was also observed that the algorithm was very sensitive to parameter values. Small variations in the initial parameter values did result in over-allocation.

RED-VBR is a method for supporting VBR video, with an off-line or on-line allocation technique. For this comparison, the on-line version was implemented using a similar segmentation algorithm as presented in [13]. RED-VBR is based upon the D-BIND model [9]. This model consists of a set of rate-interval pairs, which characterize the source over various interval lengths. The allocation algorithm stores the currently reserved D-BIND parameters and calculates the D-BIND parameters for the last M frames. A renegotiation takes place when a difference exists between the reserved and measured D-BIND parameters; more details are presented in [13]. RED-VBR does not use nor measure the QoS for allocation, since it attempts to dynamically allocate bandwidth to provide zero losses. For this reason one should expect this method to allocate higher bandwidth amounts and renegotiate more often to meet this stringent requirement. QoS is an issue when sources are multiplexed together and is

α	β	MIN_INTERVAL (seconds)	P	M
1.2	1.5	10	48	48

Table 2: RED-VBR parameter values.

provided on a "per-segment" basis as described in [13]. Since the focus of this paper is resource allocation, only the renegotiation and allocation performance of this method will be considered. The initial parameters are given in table 2. The α and β values were taken from the original paper, while MIN_REGEN_INTERVAL, P and M were selected to reduce the number of renegotiations.

Table 3 shows the performance of all the algorithms for each individual MPEG video as a source. Figure 3 shows the bandwidth allocation and cumulative CLP of all the methods for the Asterix video.

DSA+ was able to provide the desired QoS for each video, with significantly fewer bits than the peak rate. Saving of 21 - 61% were observed over peak rate. The average number of renegotiations required was 36.2 and only 44% of the renegotiations were requests for more bandwidth. On average, increases were 189 Kbps.

Hsu's algorithm was not able to provide the desired QoS for the Goldfinger, News and Lambs videos. This method also over-allocated bandwidth (more than the actual peak) for the Formula 1 Race, Mr. Bean, News, Simpsons, Super Bowl and Talk videos. This was a result of a over-allocation early in the trace, from which the algorithm was unable to reduce the bandwidth quickly enough. Placing bounds on the highest bandwidth allocated (peak) reduces this effect, but it requires the knowledge of the value *a priori*. Another difficulty with this method was the number of renegotiations. The algorithm uses constant intervals to renegotiate the bandwidth. Consequently, renegotiating every second would place a significant burden on the network's signaling system.

RED-VBR was able to provide the desired QoS for each video, with CLP values ranging from zero to 2×10^{-4}. Fewer bits than peak allocation (11 - 52% less) were used, but the algorithm required a large number of renegotiations. On average 284 renegotiations were performed, with 56% being for more bandwidth. As seen in figure 3, these increases were large, averaging 575 Kbps. The calculation of

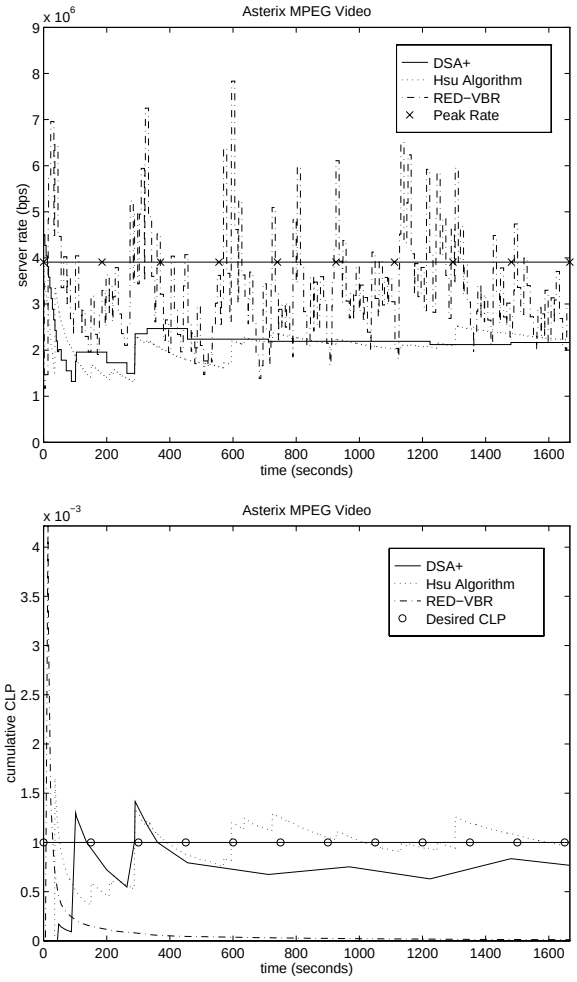

Figure 3: Bandwidth allocation and cumulative CLP for the Asterix video.

Video	DSA+		Hsu's		RED-VBR		Peak
	N.R.	Bits Used ($\times 10^9$ bits)	N.R.	Bits Used ($\times 10^9$ bits)	N.R.	Bits Used ($\times 10^9$ bits)	Bits Used ($\times 10^9$ bits)
Asterix	30	3.63	1666	3.50	305	5.77	6.51
ATP Tennis	30	5.68	1666	5.33	308	5.89	8.43
Formula 1 Race	25	5.87	1666	8.72	293	6.11	8.95
Goldfinger	47	5.11	1666	4.81	269	5.58	10.8
Jurassic Park	39	3.12	1666	2.75	296	3.89	5.28
Movie Review	35	4.38	1666	3.79	315	5.13	7.63
Mr. Bean	52	3.89	1666	13.1	269	5.03	10.1
MTV	44	6.72	1666	5.00	283	5.86	10.1
News	28	4.73	1313	9.73	220	4.64	8.60
Lambs	46	3.10	1666	2.46	279	2.82	5.93
Simpsons	36	4.56	1666	13.7	296	5.64	10.2
Soccer	25	6.53	1666	5.56	307	6.29	8.28
Super Bowl	34	4.25	1666	13.1	268	4.79	6.21
Talk	37	2.44	1666	14.6	261	3.97	4.73
Terminator	38	1.75	1666	1.44	293	2.86	3.53
Average	36.5	4.38	1666	7.17	284	4.95	7.72
Legend: **N.R.** = number of renegotiations							

Table 3: Algorithm comparison.

D-BIND parameters may also be problematic since it is done for each frame.

Overall DSA+ performed better than the other algorithms. It always required fewer bits for transmission than the peak, and on average less than the other on-line methods, while still providing the desired CLP. The significant savings was in the number of renegotiations. The algorithm required no more than 52 renegotiations and on average only 44% were for more resources. The number can be further reduced with a lower initial bandwidth value, as discussed in [6]. On average Hsu's algorithm required 47 times more renegotiations, while RED-VBR required 8 times as many. The magnitude of increases were relatively small, 189 Kbps, while RED-VBR increased three times as much. This may be problematic as contention for limited resources increases. DSA+ also has the advantage of a simple algorithm that does not require large amounts of processing time.

3.2 Network allocation

In the previous section we have only shown how to apply DSA+ for controlling the QoS of a single hop. In this section we investigate the application of DSA+ for a multiple hop connection. For these experiments four nodes are connected in series. The nodes are interconnected with 155 Mbps

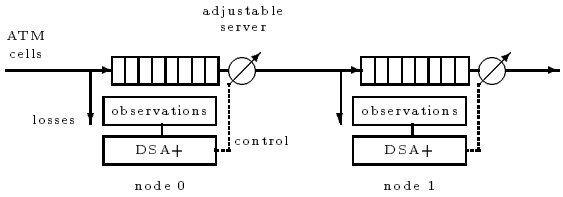

Figure 4: Multiple hop connection with each-node implementation. Only the first two nodes shown.

links, each measuring 50 meters in length. Each node consists of an adjustable rate server and finite capacity FIFO queue (80 ATM cells) as described in section 2. The MPEG video traffic entered the network at node zero and proceeded forward until node three was reached. For these experiments we are interested in providing an end-to-end cell lost probability of 1×10^{-3}. Two implementations of DSA+ were investigated: *each-node* and *first-node*.

The each-node implementation requires each node to run DSA+ separately and independently, as seen in figure 4. The end-to-end CLP was divided evenly among the nodes resulting in a target CLP of 2.5×10^{-4} per node. The end-to-end QoS could have been divided differently, perhaps based on the current condition of the individual nodes. In

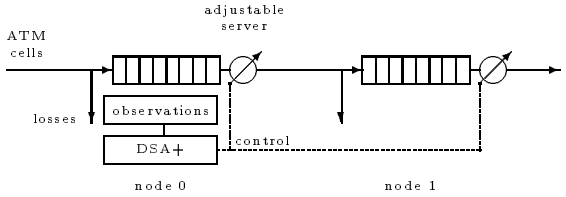

Figure 5: Multiple hop connection with first-node implementation. Only first two nodes shown.

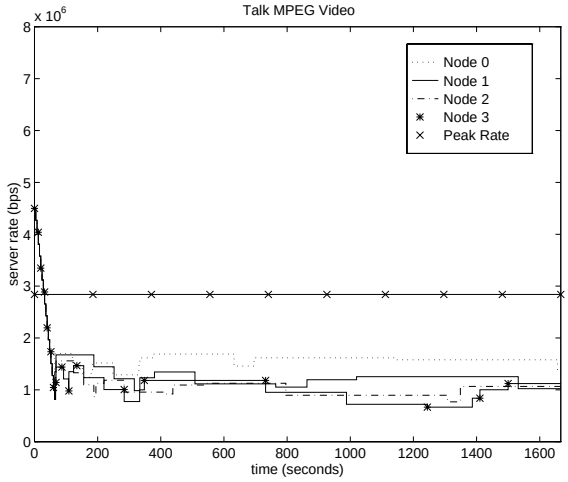

Figure 6: Bandwidth allocation with the each-node implementation.

general dividing the CLP may present problems if either there is a large number of nodes and/or if the end-to-end QoS is very stringent. The remaining DSA+ initial parameters were identical for each node and are given in table 1. One primary advantage to the strategy is that no inter-node algorithm communication is necessary, thus eliminating any need for algorithm control packets.

First-node implementation only requires the first node of the connection to run DSA+, as seen in figure 5. The initial DSA+ parameters are given in table 1. The first node controls the bandwidth for all the remaining downstream nodes. The first node has a CLP of 1×10^{-3}, therefore the remaining nodes can have zero losses. When a renegotiation occurs at the first node a control packet, containing the new bandwidth value, is sent downstream. Once a downstream nodes receives the bandwidth control packet it must immediately renegotiate to this value then forward it downstream. We assumed that the control packets are sent on another reliable connection, as done in many communication protocols. Only transmission and propagation delays were factored for the control packets.

Video	Each-node Bits Used ($\times 10^{10}$ bits)	First-node Bits Used ($\times 10^{10}$ bits)
Asterix	1.54	1.35
ATP Tennis	2.03	1.58
Formula 1 Race	2.31	2.06
Goldfinger	1.73	1.28
Jurassic Park	1.12	0.86
Movie Review	1.56	1.15
Mr. Bean	1.31	1.07
MTV	2.21	1.78
News	1.51	1.17
Lambs	1.14	0.78
Simpsons	1.56	1.27
Soccer	2.11	1.64
Super Bowl	1.39	1.06
Talk	0.85	0.71
Terminator	0.76	0.66
Average	1.54	1.23

Table 4: Multiple-hop allocation comparison.

Table 4 shows the total number of bits (summation of the bits reserved for the four nodes) reserved by each method. For the each-node method, downstream nodes required less bandwidth as seen in figure 6. This was evident for all the each-node experiments performed. This is primarily due to a reshaping effect each node has on the traffic. As the traffic passes through a node some fluctuations in the arrival stream are removed due to buffering, resulting in a less bursty departure stream. Downstream nodes benefit from this effect, resulting in a lower bandwidth allocation (1 - 47% less than the first node). The first-node implementation consistently reserved fewer total bits, as seen in table 4; yet this implementation requires the overhead of inter-node algorithm communication.

Either implementation of DSA+ for end-to-end QoS showed promising results. Since either method has some limitations, it is possible that both methods could be combined, incorporating the strengths of each.

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 band-

width of MPEG-compressed video traces with a specified allowable cell loss probability. We were interested in minimizing both the bandwidth allocated and the number of renegotiations. For the MPEG experiments fifteen actual MPEG traces were collected and used. As compared to an off-line peak-rate allocation, DSA+ saved 13–58% in bandwidth. On average 36 renegotiations were required, but only 44% were for more bandwidth, which seems acceptably low. Other methods which were compared, either over-allocated bandwidth or required up to 47 times more renegotiations.

Multiple hop connection allocation was also addressed. In this case a connection of four nodes was simulated to evaluate the performance of DSA+ for end-to-end CLP. Two implementations were investigated; each-node and first-node. Both methods were able to provide the end-to-end QoS, however each method may suffer from some possible disadvantages. More details about our work on dynamic resource allocation, including individual MPEG, allocation and CLP graphs, can be found at the web site

`ftp://ftp.csc.ncsu.edu/pub/rtcomm/rtcomm.html`

While the focus of this paper was bandwidth allocation, 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.

This paper assumed no limits on the availability of resources. When any allocation method renegotiated for more resources, they were instantly granted. However in actual implementation this assumption can not be made. In the case of network overload, where contention for more resources is high, resources may not be available. This was the primary purpose for reducing the number of renegotiations for more resource as low as possible. Nevertheless if more resources are required yet not available the users QoS will suffer. If the QoS manager has access to the MPEG compress rate, the shortage of resources can be compensated by altering the Q factor of the compression [11]. The result is a loss of picture quality, until resources are available.

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] A. Benveniste, M. Métivier, and P. Priouret. *Adaptive Algorithms and Stochastic Approximations.* Springer-Verlag, 1990.

[3] C.-S. Chang and J. A. Thomas. Effective Bandwidth in High-Speed Digital Networks. *IEEE Journal on Selected Areas in Communications*, 13(6):1091–1099, Aug. 1995.

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

[5] W.-C. Feng, F. Jahanian, and S. Sechrest. An Optimal Bandwidth Allocation Strategy for the Delivery of Compressed Prerecorded Video. To appear in *ACM/Springer-Verlag Multimedia Systems Journal.*

[6] E. W. Fulp and D. S. Reeves. Dynamic Bandwidth Allocation Techniques. Technical Report TR-97/08, Center for Advanced Computing and Communications, Aug. 1997.

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

[8] I. Hsu and J. Walrand. Dynamic Bandwidth Allocation for ATM Switches. *Journal of Applied Probability*, 33(3):758–771, 1996.

[9] E. W. Knightly and H. Zhang. Traffic Characterization and Switch Utilization using a Deterministic Bounding Interval Dependent Traffic Model. In *Proceedings of IEEE INFOCOM'95*, pages 1137–1145, Boston, MA, Apr. 1995.

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

[11] D. Reininger, G. Ramamurthy, and D. Raychaudhuri. VBR MPEG Video Coding with Dynamic Bandwidth Renegotiation. In *IEEE International Conference on Communications*, pages 1773–1777, 1995.

[12] 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.

[13] H. Zhang and E. W. Knightly. RED-VBR: A Renegotiation-Based Approach to Support Delay-Sensitive VBR Video. *Multimedia Systems*, 5:164–176, 1997.