Real-Time traffic transmission over the Internet

Marco Furini [1,2] and Don Towsley [3,4]

[1] Department of Computer Science
University of Bologna
Mura Anteo Zamboni 7 - 40127 Bologna
furini@cs.unibo.it

[3] Department of Computer Science
University of Massachusetts
Amherst, MA 01003
towsley@cs.umass.edu

Umass CMPSCI Technical Report 99-73

Abstract

Multimedia applications require the transmission of real-time streams over a network. These streams often exhibit variable bandwidth requirements, and require high bandwidth and guarantees from the network. This creates problems when such streams are delivered over the Internet. To solve these problems, recently, a small set of differentiated services has been introduced. Among these, *Premium Service* can provide low loss and low end-end delay to a real-time stream. It uses a bandwidth allocation mechanism (BAM) based on the traffic peak rate. Since the bandwidth requirement of a video stream can be quite variable, this can result in a high cost to the user and a inefficient use of network bandwidth. In this paper we consider stored video streams and introduce a BAM that can increase bandwidth utilization and decrease the allocated bandwidth without affecting the QoS of the delivered real-time stream and without introducing any modification in the Premium Service. Although the proposed BAM can substantially reduce bandwidth consumption, we introduce several frame dropping mechanisms that further reduce bandwidth consumption subject to a QoS constraint when coupled with the above BAM. The proposed BAM and the heuristics algorithms are evaluated using Motion JPEG and MPEG videos and are shown to be effective in reducing bandwidth requirements.

1 Introduction

Pay per view movies, distance learning, and digital libraries are examples of multimedia applications that require the transmission of real-time streams over a network. Such streams (such as video) can exhibit significant bit rate variability [GAR94], depending on the encoding system used, and can require high network bandwidth. Moreover these real-time streams require performance guarantees from the network (e.g., guaranteed bandwidth, loss rate, etc.). This poses

[2] The work of this author was performed while visiting the University of Massachusetts.

[4] The work of this author was funded in part by the National Science Foundation grant ANI-9805185. Any opinions, finding, or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the National Science Foundation.

significant problems when these real-time streams are delivered over the Internet, as the Internet is not able to provide any type of guarantee (it is a best effort network). Although these applications are currently deployed in the Internet, the quality of service (QoS) that they receive is far from what is desired.

There has been considerable activity recently on defining and introducing a small set of differentiated services into the Internet in order to improve the service of certain classes of applications. One such proposal by Nichols, et al. [NIC98] introduces a new service, called *Premium Service,* which can provide the QoS required by a real-time stream. Briefly, premium service provides a low loss, bounded low delay, and fixed bandwidth channel. For example, at the time that a client makes a request to a server for the play out of a video, the server would set up a premium service connection with a bandwidth equal to the peak rate associated with the video. Since the bandwidth allocation is based on the traffic peak rate, the real-time stream can be transmitted without problem because the needed bandwidth is always available. However, the peak rate allocation can be expensive in terms of bandwidth and inefficient in terms of bandwidth utilization when the video has been encoded using a variable bit rate (VBR) encoding scheme. This variability coupled with the peak rate bandwidth allocation can lead to high cost and inefficient use of bandwidth.

To reduce the variability of the traffic, smoothing techniques [SAL98] [FEN96] have been introduced. In practice, smoothing produces a new, less variable transmission of the traffic that, although having the same QoS requirement, requires less bandwidth using a peak rate allocation. Some techniques minimize the peak bandwidth requirements [FEN95]. Others [FE95b] minimize the number of rate changes and, yet, others [SAL98] reduce the variability in the rate requirements across the lifetime of the transmission plan. However, even if smoothing is used, a bandwidth allocation mechanism (BAM) based only on the traffic peak rate can still lead to low bandwidth utilization and to a waste of bandwidth.

In the past, several attempts have been made to solve the problem of inefficient bandwidth utilization. For example, [ZHI99] increases the bandwidth utilization during the play out of a video by decreasing the peak rate of the real-time stream through intelligent discarding of frames. [ZHI99] developed an optimal algorithm that minimizes the number of dropped frames given a bandwidth constraint. However, the client must settle for a lower QoS.

Another technique for solving this problem was developed in [GRO98] which introduced a mechanism called RCBR (Renegotiated CBR), that renegotiates the bandwidth during the transmission of the stream. RCBR attempts to change the allocated bandwidth as needed by the stream. However, a new problem arises when the stream requires additional bandwidth. If it is unavailable, then the stream will suffer a temporary disruption of service. This disruption of service is unpredictable and makes it difficult for the network to provide a QoS guarantee. Similar approaches have been presented in [FEN95] and [FE95b]. In the future, it may be possible to solve this problem through the advanced reservation of resources [SIK98].

The first contribution of this paper is the introduction of a new bandwidth allocation mechanism that increases bandwidth utilization and decreases the allocated bandwidth without affecting the QoS of the real-time stream delivered and without requiring any modification to the Premium Service architecture introduced in [NIC98]. For a given video stream, the BAM allocates the peak bandwidth to the premium channel. This allocation is progressively reduced as the peak rate of the remainder of the stream decreases. When coupled with optimal smoothing [SAL98], bandwidth consumption is substantially reduced even though the resulting smoothed stream doesn't look like a decreasing function.

Although the proposed BAM can substantially reduce bandwidth consumption, there may still be a further need to reduce bandwidth consumption. Our second contribution consists of several frame dropping mechanisms that further reduce bandwidth consumption subject to a QoS constraint when coupled with the above BAM. These mechanisms provide the flexibility for the client to negotiate a tradeoff between bandwidth consumption and QoS degradation with the server (and network). Using these heuristics, we show through simulation that it is possible to substantially reduce the bandwidth consumption while dropping only a few frames; depending on the movie, we can save up to 43% of the bandwidth while dropping only 1% of the frames.

The paper is organized as follow. In Section 2 we introduce our bandwidth allocation mechanism. In Section 3 we present the experimental results obtained using our BAM. In Section 4 we present the benefits in delivering slightly imperfect QoS and in Section 5 we present some heuristic algorithms for dropping some frames and the experimental results using these algorithms. The conclusions are presented in Section 6.

2 An efficient Bandwidth Allocation Mechanism

In this section we introduce a new bandwidth allocation mechanism (BAM) that increases bandwidth utilization and decreases the allocated bandwidth of a stream without affecting its QoS and without requiring any modification in the Premium Service architecture introduced in [NIC98]. Before we describe our proposed scheme, we briefly review the premium service architecture. Premium Service was introduced in [NIC98] in order to provide the QoS required by real-time streams; the server set up a premium service connection with a bandwidth equal to the peak rate associated with the video. The server can do this because the video is completely stored, and so all the video characteristics are known. Due to the variability in a video stream, this mechanism can lead to low resource utilization which may not be acceptable when someone pays for the allocated bandwidth, as is likely to happen in the coming years.

The benefits of our BAM come from the dynamic bandwidth allocation that we use during transmission. Unlike the dynamic bandwidth allocation mechanism described in [FEN97][GRO98], our mechanism provides the QoS required by the real-time stream. This is because our BAM begins by allocating the peak bandwidth to the premium channel, but progressively reduces the allocation as the peak rate of the remaining stream decreases (this progressive reduction is possible because the bandwidth characteristics of the video are known ahead of time). Consequently, there never is a need to increase bandwidth during the session. For this reason we provide the same guarantee as the classical peak rate allocation mechanism, while using less bandwidth.

Unfortunately it is very rare that the bandwidth required by the real-time stream is a decreasing function in time. For this reason our BAM is not directly based on the bandwidth requirements but is based on a *bandwidth function,* based on the traffic shape, that will be defined below.

In the following we consider a sender (that provides the service) and a receiver (that desires the service). The receiver requests a video from the sender. This video is composed of *N* frames. Without loss of generality, we assume a discrete time-model where one time unit corresponds to the time between successive frames. For a 24 frames/second full motion video, the duration of a frame is $1/24^{th}$ of a second. We denote by $a(i)$ the data sent during time i, for each $i = 1,\dots,N$.

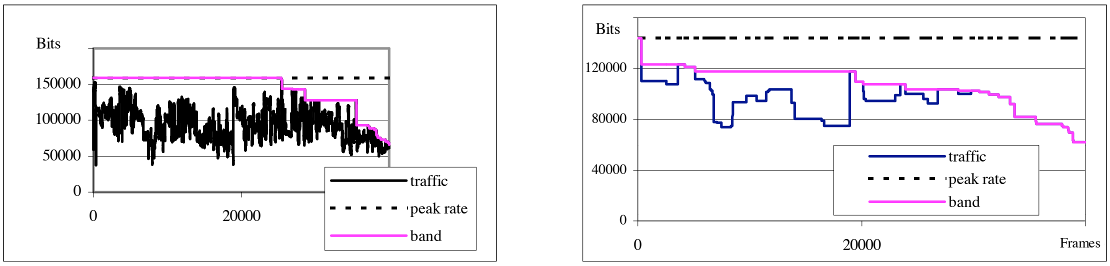
We introduce the following *bandwidth* function, which will be used by the bandwidth allocation mechanism:

$$band(i) = \max\{a(j) \mid j \geq i\} \quad i = 1,\dots,N \tag{1}$$

Here, *band*() is a non-increasing function based on the real-time stream. Since our BAM uses this function, no additional bandwidth will be requested during the transmission and it will also be possible to reduce the bandwidth when it is no longer needed: at time j (just before sending the quantity $a(j)$), a request to de-allocate the bandwidth is sent if $band(j)<band(j-1)$ and the new allocated bandwidth will be $band(j)$, instead of $band(j-1)$.

In Figure 2.1, we show the behavior of the *bandwidth* function for a real-time video stream: a Motion JPEG trace that corresponds to the movie *Big* (this and others movies will be analyzed in Section 3). Figure 2.1(a) shows the behavior of the *bandwidth* function for a pure VBR video while Figure 2.1(b) shows the behavior for the same movie, but smoothed (considering a client buffer of

1 MB and no startup delay). We note that our BAM allocates less bandwidth than the peak rate BAM because it decreases the bandwidth when future frames do not need it.



(a) (b)

Figure 2.1 Our BAM versus the classical BAM. (a) Pure VBR traffic. (b) Smoothed traffic.

The bandwidth utilization, U, achieved using our BAM is $U = \frac{\sum_{i=1}^{N} a(i)}{\sum_{i=1}^{N} band(i)}$ and is greater than what is obtained using the classic BAM because $band(i) \leq Peak\_rate \quad i = 1,\ldots,N$.

In the next section, we present experimental results that we obtained using several video traces that illustrate the benefits (like reducing bandwidth and increasing bandwidth utilization giving complete guarantees and QoS) that are possible using the proposed BAM in the premium service architecture.

3 Experimental Results

In this section we present experimental results obtained from analyzing several Motion JPEG (MJPEG) and MPEG encoded videos.

3.1 Motion JPEG

The MJPEG algorithm compresses each video frame independently using the JPEG still-picture standard. We use four different MJPEG videos [FEN95], *Big, Sleepless in Seattle, Crocodile Dundee,* and *ET,* each consisting of 40000 frames that corresponds to 28 minutes. Each video is smoothed [SAL98] for the case that the client buffer is 1 MB and zero startup delay is considered.

These experiments illustrate the benefits of our BAM by showing the reduction in the amount of required bandwidth that it introduces.

In Figure 3.1 we show the bandwidth allocation curve *band* and the bandwidth requirements for the four MJPEG videos. Also shown is the peak rate allocation. First we note, that even when the traffic is smoothed, it is still quite variable and that this variability results in the over allocation of bandwidth by a peak rate BAM. Our BAM allocates less bandwidth while still providing the same QoS because *band* better fits the traffic shape than a peak rate allocation. In Figure 3.1 (a) and (b), it is possible to note the benefits introduced by the proposed BAM in transmitting the videos *Big* and *Sleepless in Seattle*. The primary reason that these benefits are so great is that the peak rate of the smoothed traffic occurs in the initial part of the movie.

Figures 3.1(c) and 3.1(d) show the benefits of using our BAM for the movies *ET* and *Crocodile Dundee*. The benefits are less with these movies than with the previous two. This is because the peak rate occurs late in the video. The benefits of using our mechanism start around frame 30000, near the end of the video.

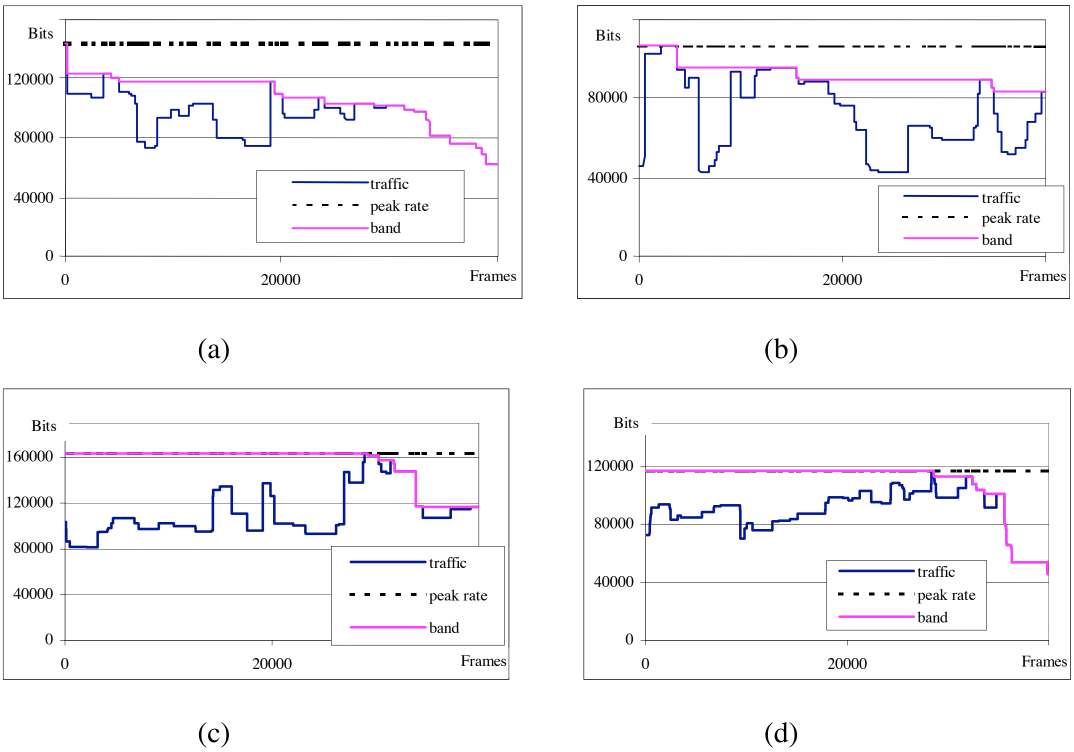
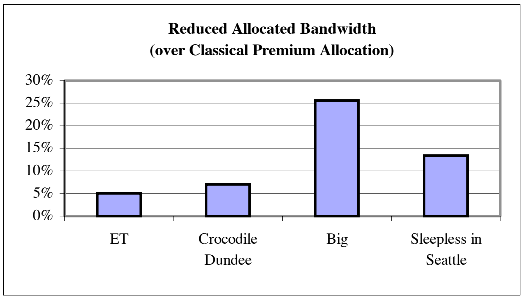
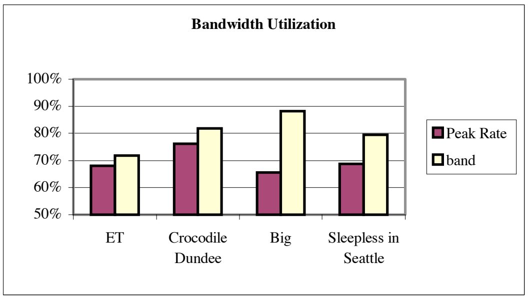


Figure 3.1. (a) *Big*. (b) *Sleepless in Seattle*. (c) *ET*. (d) *Crocodile Dundee*.

Figure 3.2 quantifies the benefits of the proposed BAM over a peak rate allocation. To better compare the two BAMs, we present in Figure 3.2(a) the bandwidth saved by implementing our BAM in the premium service architecture. We observe that the reduction in bandwidth ranges from 5% (*E.T.*) to 25% (*Big*). Figure 3.2(b) compares the bandwidth utilization achieved by the peak rate BAM and the proposed BAM. The proposed BAM always increases the bandwidth utilization. In one case, *Big*, the increase is greater than 20%. Finally, we note that the benefits are strongly correlated to the proximity of the traffic peak rate to the beginning of the video.



(a)



(b)

Figure 3.2. (a) it shows the amount of bandwidth saved if the proposed BAM is used in the premium service mechanism. (b) it compares the bandwidth utilization reached by the peak rate allocation and the proposed BAM.

3.2 MPEG traces

In this section we present results obtained from analyzing MPEG traces. MPEG is an inter-frame dependency encoding mechanism that has smaller average frame size than the MJPEG encoding. We use four different videos: *MTV, The Silence of the Lambs, Jurassic Park* and *Starwars*. All traces are 28 minutes long (except *Starwars* that is 121 minutes long), contain 12 frame GOPs and are 24 fps.

We consider a client with 1 MB of buffer available for storing video. We also considered a startup delay of 24 frames (one second). We chose this value in order to obtain considerable smoothing benefits while not incurring too long a startup delay at the client [SAL98].

Figure 3.3 shows the allocation using the classical peak rate BAM and the proposed BAM. In the case of *The Silence of the Lambs* and *MTV*, we observe (Figure 3.3(a) and 3.3(b)) that the peak rate BAM wastes a considerable amount of bandwidth while the proposed BAM allocates less bandwidth without compromising the QoS of the video delivered. The reason why the proposed BAM can save a lot of bandwidth is because the traffic peak rate is in the first half of the movie.

Although the peak rate lies in the second half of *Jurassic Park* (Figure 3.3(c)) the proposed BAM reserves a bandwidth that is very close to that required and, so, is able to reduce the bandwidth allocation to nearly the average bandwidth. With *Starwars*, Figure 3.3(d), on the other hand, the benefits are not so good because the peak rate occurs near the end of the movie, and so both the proposed BAM and the peak rate BAM allocate the same bandwidth for almost the entire movie.

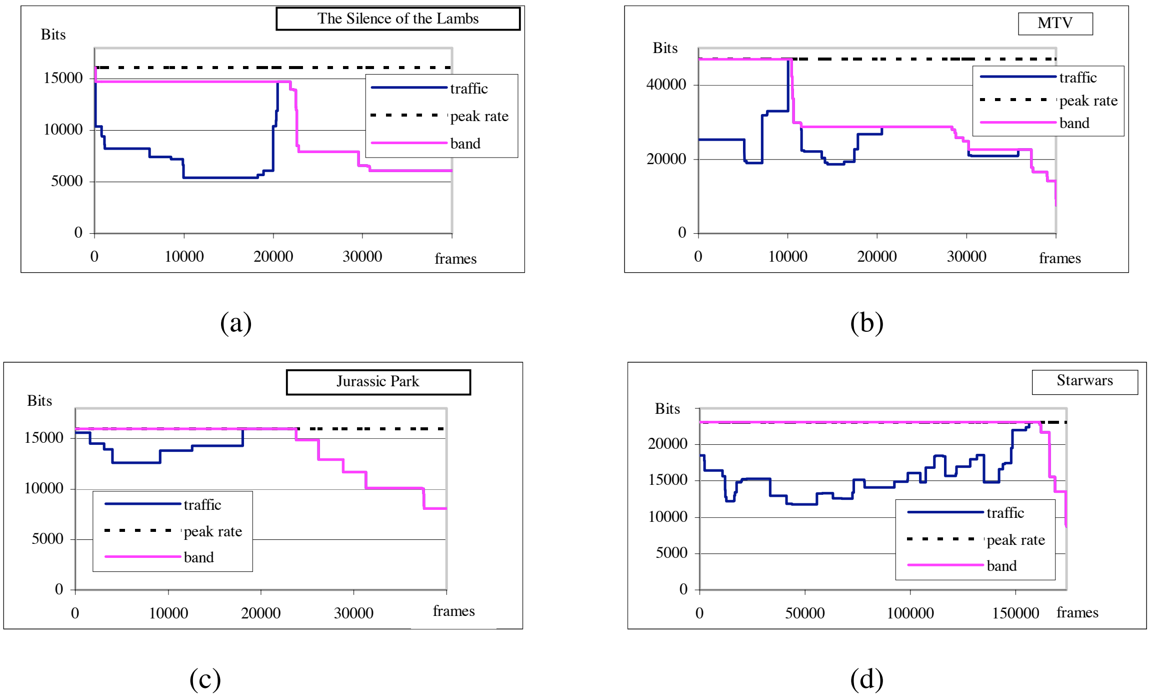
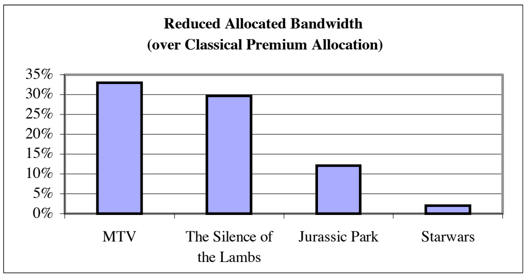
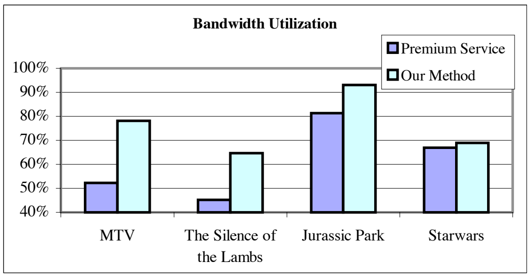


Figure 3.3. (a) *The Silence of the Lambs*. (b) *MTV*. (c) *Jurassic Park*. (d) *Starwars*.

Figure 3.4 quantifies the benefits of our proposed BAM over a peak rate allocation. The benefits introduced are remarkable since the reduction in bandwidth requests (Figure 3.4(a)) ranges from 2% (*Starwars*) to more than 30% (*MTV*). *Starwars* yields little benefit because the peak rate of the smoothed traffic occurs very close to the end of the video.



(a)



(b)

Figure 3.4. (a) It shows the amount of bandwidth that is possible to save if the proposed BAM is used in the premium service architecture. (b) It shows the bandwidth utilization reached by the proposed BAM and the peak rate BAM.

Figure 3.4(b) compares the bandwidth utilization achieved by the two mechanisms. In all of the movies analyzed, our BAM increases the bandwidth utilization from almost 30% for *MTV* to 2% for *Starwars*. The increment utilization gain obtained for *Starwars* is due to the presence of the peak rate near the end of the video.

3.3 Conclusion

In this section we presented the experimental results obtained for several video traces. We quantified the bandwidth allocated for both MJPEG and MPEG videos by the proposed BAM and the peak rate BAM. The proposed BAM can substantially reduce the required bandwidth for both types of videos without affecting the QoS. If in the coming years, the cost to the customer is proportional to the allocated bandwidth, our mechanism will result in a lower cost for transmitting the same video with the same QoS. Moreover, our BAM doesn't require any modification to the proposed premium service architecture.

4 Benefits in delivering imperfect QoS

In the previous sections we presented a bandwidth allocation mechanism for transmitting perfect QoS real-time streams such as video and illustrated its benefits over a peak rate allocation scheme. Although the bandwidth improvement can be substantial, there may be times when a client may require a further bandwidth reduction while being willing to tolerate some degradation in the QoS. Hence, in this section we present several frame dropping mechanisms which, when coupled with our BAM, provides the client and the server with the capability to tradeoff bandwidth utilization with the QoS. Imperfect QoS, does not Imply un-predictable QoS. Although less than perfect, the QoS is made known to the client. In fact, the sender and the receiver must agree on this QoS: the server and the receiver should establish a contract in which the receiver agrees to receive a defined non-perfect QoS video and the sender agrees to provide a service with that particular QoS.

Imperfect QoS means that some frames of the video are dropped. If the video is also smoothed, as recommended, the smoothed traffic presents a slightly imperfect QoS video. The client receives a continuous, although imperfect, stream and continues to play it out without halting after dealing with the imperfections through loss concealment [WZ98]. There are no changes in the BAM presented in the previous section, because the *band* function depends on the quantity of data sent at every time unit: it doesn't matter if the data represents a perfect or a non-perfect QoS.

We first begin by describing an algorithm which minimizes the peak rate of a smoothed video when the client has a constraint on the number of frames that can be dropped and the video was encoded using an intra-frame encoding algorithm. This algorithm relies on the MINFD algorithm developed in [ZHI99]. MINFD minimizes the number of frames that have to be dropped if the system has both *network bandwidth* and *client buffer* constraints. The MINFD algorithm has been shown to be optimal with respect to the minimum number of frames discarded and has complexity is O(*NlogN*). Due to lack of space we cannot explain how the MINFD algorithm works, but refer the reader can to [ZHI99] for a detailed explanation.

The problem of minimizing the peak rate subject to a constraint on the number of dropped frames can be solved by performing a binary search over possible peak rates to determine the smallest bandwidth such that the constraint on the number of discarded frames is supported. For each bandwidth value we apply MINFD to determine the minimum number of frames that need to be discarded. If it is less than the constraint, we search for a lower bandwidth; otherwise we search for a higher bandwidth. The complexity of this mechanism becomes $O(Nlog^2N)$ and the algorithm can drop the number of frames that correspond to the QoS constraint. In Figure 4.1 we present the amount of bandwidth reduction that this mechanism achieves for *Sleepless in Seattle*.

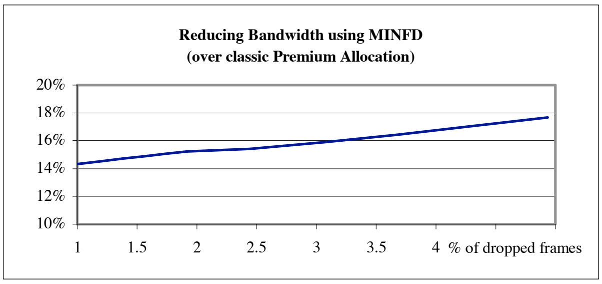


Figure 4.1 Maximum peak rate reduction as a function of percentage of frames allowed to be dropped.

Unfortunately, we are concerned with a different problem: given a specified QoS (e.g. the number of frames to drop), we are interested in reducing the *average allocated ba*ndwidth and not in minimizing the peak rate of the smoothed video. For this reason we cannot use the MINFD algorithm. Instead it is necessary to develop other algorithms that can reduce the amount of bandwidth needed for transmitting a video, given a number of frames to drop. We present some heuristic algorithm for doing this in the next section.

5 Heuristic algorithms

In this section, we present several heuristic algorithms that further try to reduce the allocated bandwidth by dropping frames from the video before smoothing it. As we stated earlier, an imperfect QoS should be used in a scenario where the server and the client agree on a particular QoS. For instance, the client could agree to receive video at an imperfect QoS if he could pay less for the same service (one can think of a distance learning systems, where the QoS of the video might not be very important). The algorithms we propose, discard a certain number of frames that correspond to the QoS established by the server and the client (let us denote this number by k) in order to reduce the bandwidth needed for transmitting the video. We develop separate sets of heuristic algorithms for MJPEG and MPEG.

5.1 Motion JPEG

Discard Largest Frames (DLF): The DLF policy discards the largest k frames of the video. The algorithm is very simple and has computational requirements that are linear in N. This algorithm may discard consecutive frames, which may lead to a poor quality playback of the video at the receiver side.

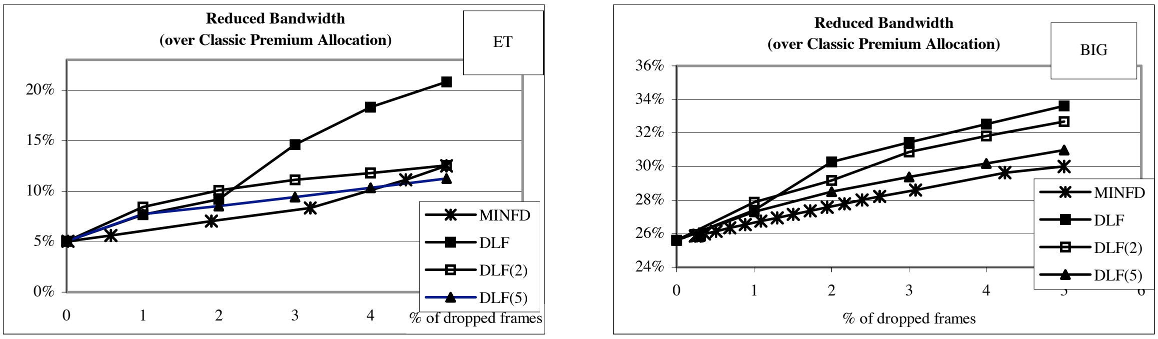
Discard Largest Frame with distance λ (DLF(λ)): DLF(λ) is a variation of the previous algorithm. Again, DLF(λ) discards the largest k frames of the video, but it discards a frame only if the distance from the previous discarded frame is equal or greater than λ. In this way, DLF(λ) maintains a minimum distance of λ frames between two consecutive dropped frames. Again, this algorithm is very simple to implement.

5.1.1 Experimental results

Our experiments compare the heuristic algorithms (whose purpose is to reduce the average allocated bandwidth) with the mechanism, described in the previous section, which uses MINFD to minimize the bandwidth peak rate. We use three algorithms: DLF, DLF(2) and DLF(5) on the MJPEG traces we used in Section 3, in order to compute the amount of the reduced bandwidth reached by these algorithms. Figures 5.1 and 5.2 illustrate the bandwidth reduction over a peak rate allocation as a function of the number of dropped frames. We observe that our algorithms result in lower bandwidth consumption than MINFD. This is because the mechanism that uses MINFD minimizes the bandwidth peak rate, while our heuristic algorithms are concerned with reducing the average allocated bandwidth. In all of the experiments, we observe that discarding a small percentage of frames can greatly reduce the allocated bandwidth.

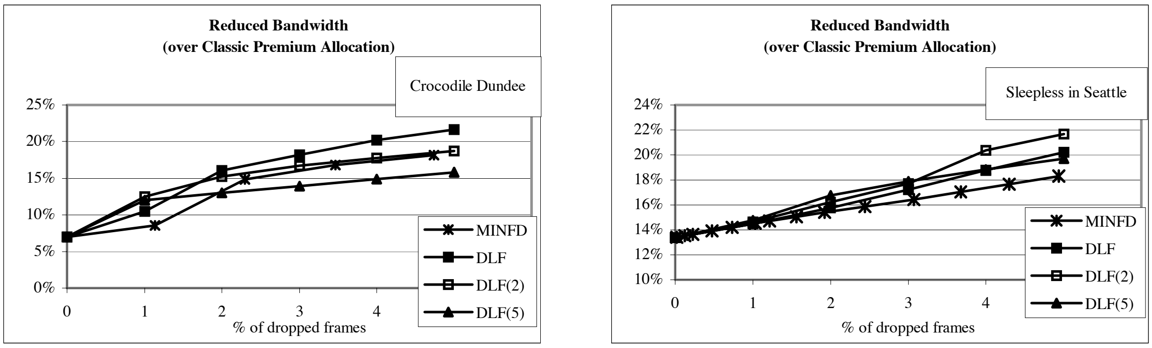
In Figure 5.1(a) (*ET*) there is little difference between the performance of the heuristics when the percentage of dropped frames is less than 3%. If a larger percentage of dropped frames is permitted, then DLF reduces considerably more bandwidth than the heuristics. The benefits in this case are quite good because dropping 5% of the frames results in an additional reduction of 17%. In Figure 5.1(b) we show the results for *Big*. The algorithms differ slightly and they also permit a further reduction of the allocated bandwidth from 26% (perfect QoS delivered) to 33% (5% of dropped frames).

In Figure 5.2 (a), discarding 2% of the frames results in a 16% reduction in the allocated bandwidth over a peak rate allocation. All three discard policies perform similarly with DLF slightly better than DLF(2) which is slightly better than DLF(5) as a greater percentage of frames are allowed to be discarded. This is because DLF can discard consecutive frames, while DLF(2) and DLF(5) cannot. For *Sleepless in Seattle* (Figure 5.2(b)) the three algorithms produce almost the same results, a reduction in the allocated bandwidth from 13% (perfect QoS delivered) to 22% (5% of dropped frames).



(a) (b)

Figure 5.1. (a) *ET*. (b) *Big*.

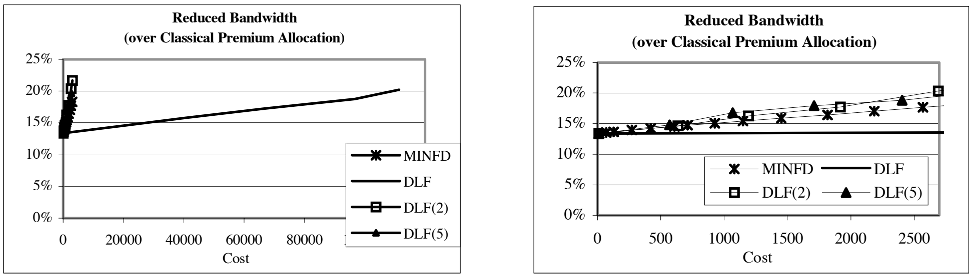


(a) (b)

Figure 5.2. (a) *Crocodile Dundee*. (b) *Sleepless in Seattle*.

In all of the experiments, DLF performs better than the other two heuristics (an exception is *Sleepless in Seattle*, where the algorithms produce almost the same results). This can be explained by observing that DLF can drop consecutive frames and, consequently can drop blocks of the video: in the worst case all of the discarded frames could be consecutive frames and this could mean bad playback quality.

So far we have focused on the number of dropped frames. Unfortunately there may not be much correlation between this measure of QoS and what the user perceives. Hence we focus on a cost function introduced in [ZHI99] which attempts to penalize algorithms that drop neighboring frames. This cost function takes two aspects into consideration: the length of a sequence of consecutive discarded frames and the distance between two adjacent but non-consecutive discarded frames. This cost function assigns a cost c_j to a discarded frame j depending on whether it belongs to a sequence of consecutive discarded frames or not. If frame j belongs to a sequence of consecutive discarded frames, the cost is l_j, if the frame j is the l_j^{th} consecutively discarded frame in the sequence. Otherwise the cost is given by $1+1/\sqrt{d_j}$, where d_j represents the distance from the previous discarded frame. More details about this cost function can be found in [ZHI99]. We present the cost achieved by the heuristic algorithm when applied to *Sleepless in Seattle* in Figure 5.3: it shows that DLF performs completely worse that the others. We observe in Figure 5.3(b) that DLF(2), DLF(5) and MINFD achieve very similar results.



(a) (b)

Figure 5.3. (a) Reduced bandwidth in function of the cost. (b) An enlargement of (a).

Based on this cost function, DLF is not worth using because its cost is too high and because its performance (see section 4.1.1.) is almost like that of DLF(2) and DLF(5). These two algorithms have very similar values both for the cost and for the reduced bandwidth. Since DLF(2) results in a greater bandwidth reduction, it is preferred to DLF(5).

5.2 MPEG

In the previous section we proposed some heuristic algorithms for MJPEG video. Since the MPEG encoding differs from the MJPEG encoding, we cannot use the previous algorithms for MPEG video. For this reason, in this section, we present some heuristic algorithms specifically designed for MPEG video. In a MPEG video, the frames don't have the same importance as some frames depend on other frames. We use MPEG videos organized in Groups Of Picture (GOP) with a size of 12 frames. MPEG can use three types of frames: *I, P* and *B* frames. The GOP is composed as: $IB_1B_2P_1B_3B_4P_2B_5B_6P_3B_7B_8$. To decode a *B* frame, both the previous and future *I* or *P* frames are needed. To decode a *P* frame the previous *P* or *I* frame is needed. An *I* frame can be decoded without the use of any other frames. Thus, discarding an *I* frame results in the loss of an entire GOP (plus the two B frames of the previous GOP that depend on the I frame discarded) while the discard of a P frame results in the loss of the P and B frames that depend on it. In summary, the discard of the I frame results in the discard of 14 frames, the discard of the P_1 frame results in the discard of 11 frames, the discard of the P_2 frame results in the discard of 8 frames, and the discard of the P_3 frame results in the discard of 5 frames. Only the discard of a B frame results in no additional frame discard.

Based on these dependencies we propose the following algorithms:

Drop I Frame (DIF): DIF drops the largest GOPs (plus the two B frames preceding the I frame) of the video. If the maximum number of GOPs that can be discarded is denoted by L, the algorithm selects the L largest GOPs of the video and discards them. Hence, if k is the maximum number of frames that can be dropped, $L = k/14$.

DIF(λ): this is a variation of the previous algorithm which considers not only the size of the GOP, but also the distance from the previous discarded GOP. The algorithm, uses λ as a parameter that indicates the minimum distance between discarded GOPs (here distance is expressed in GOP).

Discard First P Frame (DFPF): DFPF discards the largest groups of frames that depend on the P_1 frame. In this case, the discarded group is composed by 11 frames of the GOP (only the I frame of the GOP is not discarded, since it doesn't depend on any other frames of the GOP). Again, if L is the maximum number of these P_1 group that can be discarded, the algorithm orders the P_1 group and it discards the L largest P_1 groups. In this case $L = k/11$.

Discard Second P Frame (DSPF): DSPF discards the largest group of frames that depend on the P_2 frame. In this case, 8 frames compose the discarded group. Hence, $L = k/8$.

Discard Third P Frame (DTPF): DTPF discards the largest group of frames that depend on the P_3 frame. 5 frames, in this case, compose the group. Hence, $L = k/5$.

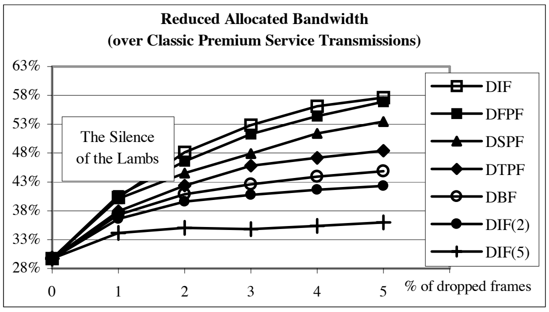
Discard B Frame (DBF): DBF discards only the B frames of the video. If L is the maximum number of B frames that can be discarded, the algorithm orders the B frames and discards the largest L frames. ($L = k$).

In our experiments we use the same MPEG videos of Section 3. We use DIF, DIF(2), DIF(5), DFPF, DSPF, DTPF and DBF in order to compute the amount of reduced bandwidth achieved by these algorithms. Figures 5.4 and 5.5 illustrate the bandwidth reduction over a peak rate allocation as a function of dropped frames. We observe that discarding a small percentage of frames can greatly reduce the allocated bandwidth.

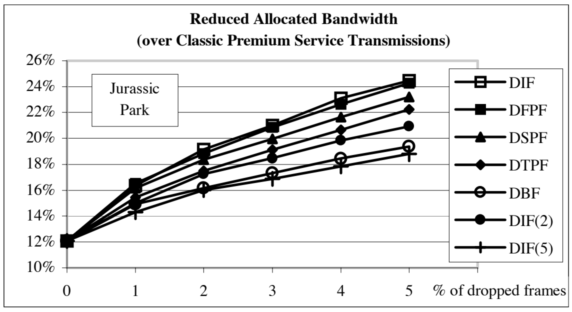
In Figure 5.4(a) we present the results obtained from analyzing *The Silence of the Lambs*. With this trace DIF and DFPF achieve great results since a drop of 5% of the video results in reducing the bandwidth by 58%. Note that a drop of 1% allows a bandwidth saving of up 42%.

Figure 5.4(b) shows the results obtained from analyzing *Jurassic Park*. Again, a drop of 1% of the video results in a bandwidth reduction of 16%. DIF and DFPF result in a bandwidth saving of 24% when 5% of the video is dropped.

In Figure 5.5(a) we present the results obtained from analyzing *Starwars*. A drop of 1% of the video allows a bandwidth saving of 7% and a drop of 5% of the video results in a bandwidth reduction of 22% (using DIF). Figure 5.5(b) shows the results obtained from analyzing *MTV*. Once again, our heuristic algorithms allow a bandwidth saving of 48% when 5% of the video is dropped and a drop of 1% of the video results in a bandwidth reduction of 43%.

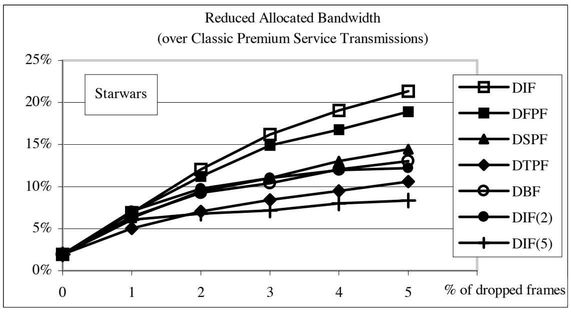


(a)

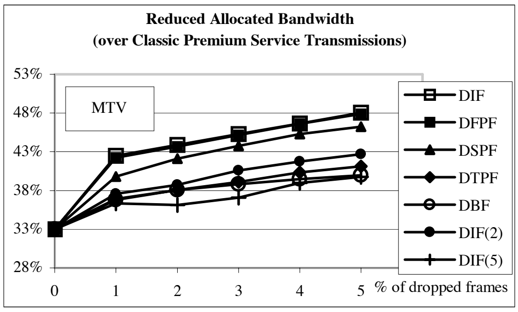


(b)

Figure 5.4. (a) *The Silence of the Lambs* (b) *Jurassic Park.*



(a)



(b)

Figure 5.5. (a) *Starwars*. (b) *MTV*.

6 Conclusions

In this paper we presented a new bandwidth allocation mechanism that can be used in the premium service architecture introduced in [NIC98] to handle real-time variable bit rate streams over the Internet. We developed it in order to increase the bandwidth utilization for a stream under the premium service architecture. Our BAM is easily implemented in the premium service architecture, as it doesn't require any architectural modification. We show, through several experiments, that its use can greatly reduce the allocated bandwidth for transmitting the same traffic with the same QoS and the same guarantee. The experiments show that our BAM can reduce by up to 30% of the bandwidth needed for transmitting the video.

We showed further benefits possible by sending slightly imperfect QoS video. We developed some heuristic algorithms that can be used to drop frames in order to minimize the bandwidth consumption. All of the experiments presented in this paper show that the use of our algorithms can possibly lead to a great reduction in bandwidth while dropping very few frames. For instance, it is possible to save up to the 43% of the bandwidth dropping only the 1% of the video for *MTV*; up to 42% dropping 1% of *The Silence of the Lambs*.

Since bandwidth is a precious resource, we think that the proposed mechanisms may be very useful both for the server and the client. The client could be happy to pay less for the same service or for a slightly imperfect service and the server could be happy because reducing the bandwidth needed for one service could mean making bandwidth available for others services.

Our study has assumed knowledge of the bandwidth characteristics of the video stream. This is reasonable in the case of stored video. We are investigating the problem of handling video streams that are generated on-line in order to remove this constraint. Further, we are studying how to modify our BAM in order to introduce interactive controls (like the VCR functions) in the mechanism.

7 References

[FEN95] W.C.Feng, S.Sechrest, "Critical bandwidth allocation for the delivery of compressed video", Comp. Comm. Vol.18, pp.709-717, Oct. 1995.

[FE95b] W.C.Feng, F. Jahanian, S.Sechrest, "Optimal buffering for the delivery of compressed prerecorded video", in Proc. Of the IASTED/ISMM Int'l Conf. On Networks, Jan. 1995.

[FEN96] W.C.Feng, "Video-on-Demand Services: Efficient Transportation and Decompression of Variable Bit Rate Video", Ph.D. Thesis, Univ. of Michigan, April 1996.

[FEN97] Wu-chi Feng, J. Rexford, “A Comparison of Bandwidth Smoothing Techniques for the Transmission of Prerecorded Compressed Video”, IEEE INFOCOM 1997, Kobe, Japan, pp. 58-66, April 1997.

[GAR94] M. Garret and W. Willinger, “Analysis, Modeling and Generation of Self-Similar VBR Video Traffic”, Proc. ACM SIGCOM, pp. 269-280, Aug. 1994.

[GRO95] M. Grossglauser, S. Keshav and D. Tse, “The Case Against VBR”, Proc. 5th Intl. Workshop on Network and Operating System Support (NOSSDAV '95), Durham, N.H., April 1995.

[GRO98] M. Grossglauser, S. Keshav, and D. Tse, “RCBR: A Simple and Efficient Service for Multiple Time-Scale Traffic”, to appear in IEEE/ACM Trans. on Networking, 1998.

[NIC98] K, Nichols, V.Jacobson, L.Zhang, "A Two-bit Differentiated Services Architecture for the Internet", Internet Draft, 1998.

[SAL98] J. D. Salehi, Z.-L. Zhang, J. F. Kurose, D. Towsley, “Supporting stored video: Reducing rate variability and end-to-end resource requirements through optimal smoothing”, IEEE/ACM Trans. Networking, vol. 6, pp. 397-410, August 1998.

[SIK98] J. Sikora, B. Teitelbaum, “Differentiated Services for Internet2”, http://www.internet2.edu/qos/may98Workshop/html/diffserv.html

[WIJ96] D. Wijesekera, J. Srivastava, “Quality of Service (QoS) Metrics for Continuos Media”, Multimedia Tools and Applications, Vol. 2, No.3, pp. 127-166, Sept 1996.

[WZ98] Y. Wang, Q. Zhu, “Error Control and Concealment for Video Communications: A Review”, Proc. of the IEEE, Vol. 86, pp. 974-997, May 1998.

[ZHI99] Z.-L. Zhang, S. Nelakuditi, R. Aggarwal, R. P. Tsang, “Effcient Selective Frame Discard Algorithms for Stored Video Delivery across Resource Constrained Networks”, To appear in Proc. IEEE INFOCOM'99, NYC, March 1999.