16

A Guaranteed-Rate Channel Allocation Scheme and Its Application to Delivery-on-Demand of Continuous Media Data[1]

T. Kameda[a], J. Ting[b] and D. Fracchia[c]

[a]School of Computing Science, Simon Fraser University, Burnaby, B.C., Canada V5A 1S6

[b]Bell Northern Research, P.O.Box 3511, Station C, Ottawa, Ontario, Canada K1Y 4H7

[c]School of Computing Science, Simon Fraser University, Burnaby, B.C., Canada V5A 1S6

Abstract

We propose a new reservation-based bandwidth allocation method to satisfy the requirements of real-time data. It reserves bandwidth in the form of uniformly distributed time slots on a link. It is conceptually simple and should be easy to implement. Delay-insensitive or loss-insensitive data can be transmitted concurrently, using the unreserved bandwidth.

As an example of its application, we discuss the on-demand delivery of pre-recorded, continuous media data. Given the file size, the playback rate, and the buffer space available at the client site, we compute the range for data transmission rate that prevents client buffer underflow and overflow. Based on this range, our channel allocation scheme can reserve bandwidth along the connection from the server to the client. We first discuss in detail the case where data are not compressed, so that all playback-frames of data consist of the same number of bytes, and then report similar results on the compressed data case, in which different playback-frames may consist of different numbers of bytes.

Keyword Codes: C.2.3; H.3.5; H.5.1
Keywords: Network Operations; Online Information Services; Multimedia Information Systems

1. INTRODUCTION

1.1 ATM and Real-Time Applications

With the recent advances in digital communication technology, it has become possible to transmit many different types of information in digital form in an integrated network, sharing the network resources. The *Asynchronous Transfer Mode* (ATM) of *Broadband Integrated Systems Digital Network* (B-ISDN) [11, 19] is intended to take advantage of the reliability and fidelity of modern digital communication facilities, in order to provide much faster packet switching than has been available in the past. ATM has often been

[1]This work was partially supported by grants from the Natural Sciences and Engineering Research Council of Canada and the Advanced Systems Institute of British Columbia.

G. Neufield et al. (eds.), *Protocols for High Speed Networks IV*

hailed as a panacea for all communication needs. However, it's been realized that, by its very nature, it is not straightforward for ATM to deliver the kind of quality-of-service required by many real-time applications (e.g., [3, 7, 9]). A great deal of effort has recently been made to address this problem (e.g., [7, 10, 13, 20]). We propose a new bandwidth allocation scheme, which represents our contribution towards solving this problem.

Although our allocation scheme has a wide range of applications, we focus on the delivery-on-demand of pre-recorded, continuous media data. It is envisioned that consumers will be able to select video programs in the comfort of their home, instead of renting video tapes from a video rental store. The simplest method for delivering a movie would be to make the client wait until the entire movie had been transmitted to the buffer of the client's machine before viewing the movie. This approach is not satisfactory if the client does not want to wait that long, or there isn't enough storage in the client's machine to store the entire movie. In such situations, we need to start viewing the movie before the data transfer is complete.

In this paper, we discuss in some detail:

1. a new bandwidth allocation scheme for a communication link that meets strict real-time requirements without packet loss, and
2. its application to the on-demand delivery of pre-recorded, continuous media data (compressed at variable rate) for real-time playback, when the client has a finite storage capacity.

1.2 Previous Work

In addition to those contained in the papers we cited above, a number of other solutions have been proposed for the bandwidth allocation for real-time data transfer in delay-sensitive packet-switching networks [4, 5, 14, 15, 16], and storage and retrieval of multimedia information [2, 6]. However, even in the absence of failures and errors, meeting stringent real-time deadlines with packet-switching is difficult, and most of the solutions cannot offer 100% guarantees.

In most real-time applications, end-to-end delay and jitter are probably the most important quality-of-service parameters, while minimizing the amount of buffer required inside the network is perhaps of secondary importance. However, depending on the application, the relative importance of these parameters varies. For one-way, real-time video or voice transmission, for example, the absolute amount of delay is, in most cases, less important than the jitter. In this section, we first review some representative bandwidth allocation and scheduling schemes to put our contribution in perspective. Based on a recent paper by Li, et al. [10], we compare several schemes with respect to three criteria: (i) maximum end-to-end queueing delay (excluding the propagation delay), (ii) maximum jitter, and (iii) the worst case buffer requirement at a network node. In Table 1 below, the following parameters are used.

- B – Link bandwidth (bits/sec)
- r_m – smallest connection rate (the transmission rate of the narrowest bandwidth assigned to a connection)
- h – route length (in number of hops)

	Max queueing delay	Jitter	Buffer required per node
Stop-and-Go[7]	$2h(l/r_m)$	$h(l/r_m)$	$2B(c/r_m)$
VirtualClock[20]	$h(l/r_m)$	$h(l/r_m)$	$B(c/r_m)$
Immediate forwarding[10]	hT_f	T_f	BT_f
2-frame choice[10]	$2hT_f$	hT_f	$2BT_f$
Binary Preallocation	$h(c/B)$	0	c

Table 1: Comparison of various schemes
(All but the last row is from [10].)

- c – cell size (in ATM, $c = 53 \cdot 8 = 428$ bits)
- l – cell payload size (in ATM, $l = 48 \cdot 8 = 384$ bits)
- T_f – frame time

ATM service is based on the transmission of small, fixed-size packets called *cells*. In this paper, we use the terms, cell and packet, interchangeably.

We refer the reader to the discussion of the first four rows of the above table to reference [10]. It is commented in [10] that *Stop-and-Go queueing* has a limitation in that its frame time (or scheduling interval) must be fixed at c/r_m, which in turn determines the maximum queueing delay for all connections. If a voice connection is to be made at 64Kb/sec through an ATM network, for example, $l/r_m = (48 \cdot 8/64 \cdot 10^3)sec = 6msec$, so that the maximum queueing delay per node is $12msec$. Delays of this magnitude may be intolerable for other applications with higher-speed connection rates.

The basic idea of the *Immediate forwarding* scheme is that since the maximum queueing delay, for example, is proportional to the frame time T_f, T_f should be kept small, independently of the desired minimum connection rate, r_m. If r_m satisfies $kT_f = c/r_m$, then the minimum-rate connection can use every kth frame to transmit its cells.

The last row of the above table summarizes the properties of *Binary Preallocation* that we propose in Section 3 of this paper. They are derived based on the assumption that all the local clocks of the nodes tick at exactly the same rate but they are not synchronized with each other [8, 12], so that the cell slots of an input link are not necessarily aligned with those on the corresponding output link. If they were perfectly aligned, taking the cell processing time at the node into account, then both the max queueing delay and buffer required per node would be 0 (the queueing delay does not include the cell processing time). Another important feature of our scheme is that it imposes no lower bound, r_m, on the available connection rate, so that an arbitrarily low connection rate can be provided without causing any wasted bandwidth.

Gemmell [6] discusses the retrieval from disk and playback of continuous media data. His model (which doesn't include a network) is developed specifically for the real-time requirements of digital audio playback. He shows how to describe these requirements in terms of the consumption rate and the data-retrieval rate. Lower bounds are derived on buffer space for certain common retrieval scenarios. Gemmell's work motivated us to investigate a more general case, where there is a network between the source and the playback unit. When pre-recorded, compressed video data are transmitted over a network

and played back at the destination, a variable number of cells are consumed per playback-frame. To analyze such a system, we use a model in which the playback unit has a variable playback rate within a certain range.

The rest of the paper is organized as follows. In the next section, we present our model and define some useful terms. In Section 3, we propose a new bandwidth allocation scheme, called *Binary Preallocation*, mentioned above. The remaining sections discuss an application of *Binary Preallocation* to the on-demand delivery of pre-recorded, continuous media data. In Section 4, we first consider the buffer overflow and underflow problems for uncompressed data, and derive tight lower bounds on the required buffer size. Based on these bounds, we then derive the feasible range for the data transfer rate. Section 5 will investigate those problems for compressed data.

2. THE MODEL

We model a network by a directed graph, in which the directed edges represent unidirectional communication links. A server is represented by a node of outdegree 1 and indegree 0, and a client is represented by a node of outdegree 0 and indegree 1. All other nodes participate in transmitting packets (cells).

To model ATM, we assume that all cells sent over the network have the same size, and the bandwidth of a communication link is expressed in terms of the number of cells it can transmit per second. The unit of playback is called a *playback-frame*, and it consists of a sequence of cells. We further assume that the capacity of a link ℓ is of the form $R_0/2^{h(\ell)}$ (cells/sec), where R_0 (cells/sec) is a constant for the network but the non-negative, dimensionless integer $h(\ell)$ depends on the particular link ℓ.

Each request by a client is characterized by the following three parameters.

1. F: the file size (in the number of cells it consists of).
2. B: the buffer space (in the number of cells that it can store) available for this request at the client's site.
3. R_c: the *consumption* (or *playback*) *rate*, i.e., the rate of playback in cells/sec.

A service for a multimedia on-demand application can be provided in three phases: the *connection set-up, data transmission* and *clear-up* phases. We now elaborate on the connection set-up phase. Upon receiving a request from a client, a network node calculates the feasible transfer rate range and then forwards the request to the appropriate server. The network then uses a routing algorithm to find a route between the server and the client, which is called a *global logical channel* or *connection*. Every link on this connection has a *local logical channel* or, simply, a *channel*, corresponding to this global logical channel. To ensure that there are enough resources in every node (processing capacity and buffer space) and link (bandwidth) along a global logical channel, a routing algorithm performs an *admission test* at each node along the path it is trying to find. If the admission test succeeds, the required resources at the node are temporarily reserved for the request. Finally, if a feasible connection is found, the network accepts the request and proceeds to the data transmission phase. Otherwise, the request is rejected, and the temporarily reserved resources are released.

If the delivery rate of playback-frames from the network is lower than the consumption rate, R_c, playback-frames must first be prefetched and buffered, before playback can start.

We assume that, while playback is in progress, one playback-frame in the buffer is used exclusively for playback and no part of it should be modified. If the buffer does not hold the next playback-frame when the playback of the current playback-frame is complete, playback is disrupted. We call this problem the *buffer underflow problem.* On the other hand, if the playback-frame delivery rate is higher than R_c, unless enough buffer space is provided, the arriving cells may overflow the buffer before they are played back. We call this the *buffer overflow problem.* Gemmell [6] shows that starting playback as soon as possible minimizes the required buffer space and vice versa.

In the rest of this paper, a frame will mean a playback-frame (i.e., not the time frame we discussed in the previous section).

3. BANDWIDTH ALLOCATION

Given that we want to use ATM for all applications, the problem we face is how to support synchronous applications within the framework of ATM.

We divide the time on each link into *cell slots*, whose size is the time needed to output a single cell over that link. Suppose link ℓ can transmit $R(\ell) = 2^{-h(\ell)}R_0$ cells/sec. Number the cell slots of link ℓ as 0,1,2, ... We define the subchannel consisting of the even-numbered cell slots, i.e., 0, 2, 4, ..., and call it C_0^ℓ. So, for $i = 0, 1, 2, \ldots$, the ith cell slot belonging to C_0 (we omit the superscript ℓ, when it is clear from the context) is given by

$$C_0(i) = 2i.$$

Similarly, C_1 consists of the odd-numbered cell slots, i.e.,

$$C_1(i) = 2i + 1.$$

In general, the subscript α in C_α is a binary string whose length is denoted by k, i.e., $\alpha = d_1 d_2 \cdots d_k$. When $k = 0$, $\alpha = \Lambda$ (the null string). Formally, C_α can be defined recursively as follows: The logical channel consisting of the total bandwidth of link ℓ, i.e., C_Λ^ℓ, is defined by

$$C_\Lambda(i) = i.$$

Given $C_{d_1 d_2 \cdots d_{k-1}}$,

$$C_{d_1 d_2 \cdots d_{k-1} 0}(i) = \text{ the } i\text{-th even cell slot in } C_{d_1 d_2 \cdots d_{k-1}} = C_{d_1 d_2 \cdots d_{k-1}}(2i), \tag{1}$$

and

$$C_{d_1 d_2 \cdots d_{k-1} 1}(i) = \text{ the } i\text{-th odd cell slot in } C_{d_1 d_2 \cdots d_{k-1}} = C_{d_1 d_2 \cdots d_{k-1}}(2i + 1). \tag{2}$$

From the above definition, we can derive the following formula.

$$C_{d_1 d_2 \cdots d_k}(i) = 2^k \cdot i + d_1 + d_2 \cdot 2 + \cdots + d_k \cdot 2^{k-1}. \tag{3}$$

Binary Preallocation Algorithm (BPA)

Let R (cells/sec) be the requested bandwidth on link ℓ.

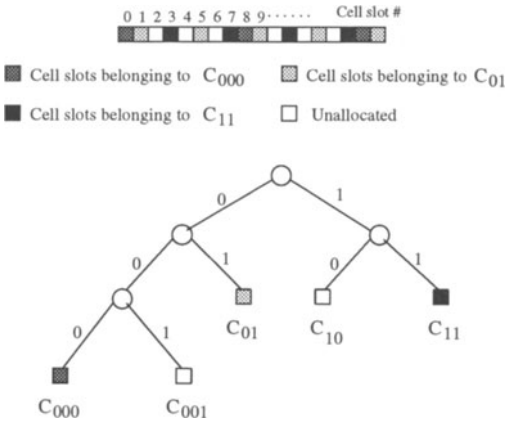

Figure 1: *Binary Preallocation* and its tree representation.

1. Find an unallocated (potential) channel, $C_{d_1 d_2 \cdots d_{j-1}}$ with the minimum transmission speed such that $R(\ell)/2^{j-1} \geq R$. Let $k = j$.
2. **while** $R(\ell)/2^k \geq R$ **do**
   divide the channel into two subchannels (each with the transmission speed of $R(\ell)/2^k$ cells/sec) according to the formulas (1) and (2) and pick one of them. Let $k := k+1$.
   end; □

BPA allocates the channel found in Step 1 to the request, if the while loop in Step 2 is executed 0 times; otherwise, it allocates the channel picked in Step 2. Fig. 1 depicts the result of applying BPA to three requests for bandwidths of $R(\ell)/8$, $R(\ell)/4$ and $R(\ell)/4$, respectively.

Comments:

1. Let ℓ and ℓ' be two links meeting at a node with bandwidths $2^{-h(\ell)}R_0$ and $2^{-h(\ell')}R_0$, respectively, where $h(\ell') \geq h(\ell)$. Then for an arbitrarily chosen channel for ℓ', BPA always finds a channel of ℓ with the same speed, provided there is enough unallocated bandwidth left on ℓ. Thus, we can establish a global logical channel out of local logical channels of the same bandwidth.
2. For a high-speed network, the switching (at the intermediate nodes) of cells belonging to the same connection would be implemented by hardware. The subscript α of the allocated channel C_α can be used as the control input to this switching hardware.
3. BPA has some resemblance to the "buddy system" of main memory allocation used in some operating systems. Therefore, the book-keeping of allocated and unallocated logical channels can be done in a similar way. For example, if two logical channels $C_{d_1 d_2 \cdots d_{k-1} 0}$ and $C_{d_1 d_2 \cdots d_{k-1} 1}$ are released, then they should be combined into one logical channel, $C_{d_1 d_2 \cdots d_{k-1}}$, for future allocation. Also, the unallocated logical

channels can be kept track of by placing them in different lists depending on their bandwidth.

Advantages:

1. Little buffer space is required (see Table 1) at each network node along the connection.
2. Scheduling at each network node is very simple. (See Comment 2 above.)
3. The cells on different connections do not interfere with each other, so that no congestion will result.
4. The unused bandwidth is always in the form of cell slots which are uniformly distributed over time. Thus, periodic cell slots can be allocated to future requests.
5. Unlike the other schemes discussed in Section 1, a connection with an arbitrarily low rate can be established without causing any wasted bandwidth.
6. Two types of synchronized media, e.g., voice and video, could be transmitted using two separate *virtual channels* in the same *virtual path* in an ATM network. The voice and video cells to be synchronized with each other arrive at the client within a fixed time bound.

Disadvantages:

1. Unallocated bandwidth may become fragmented. Thus, no single unallocated channel may be able to satisfy a new request, even though there is still enough unallocated bandwidth. Unlike memory allocation, however, fragmented bandwidth need not lead to waste. Two or more unallocated channels whose total bandwidth is not less than the required bandwidth can be allocated to satisfy a request.
2. Dedicating a fixed amount of bandwidth is wasteful for bursty real-time data transmission. This problem can be mitigated to a certain degree by the following approach, which we call *delayed forwarding.* (See Fig. 2. In the figure, the channel designators, α and α', have the same length, so that C_{α}^{l} and $C_{\alpha'}^{l'}$ have the same speed.) When cells belonging to a connection arrive on an input link of a network node, the node holds each cell for a fixed interval of time before forwarding it on the output link for the connection. This will enable the node to discover the unused cell slots and use them for other traffic. This approach works, of course, provided this delay is acceptable for the application.
3. The requested bandwidth R may not be equal to $R(\ell)/2^k$ for any integer k, so that a part of the allocated bandwidth $(R(\ell)/2^k - R)$ is wasted. This problem can be overcome by one of the following two methods: (i) Instead of allocating $R(\ell)/2^k$ to the request, allocate a set of channels including one with bandwidth $R(\ell)/2^{k+1}$ and some others with smaller bandwidths such that the total bandwidth is very close to R. (ii) *Delayed forwarding* discussed above.

The cell slots belonging to unallocated channels should be used for other delay-insensitive or loss-insensitive data.

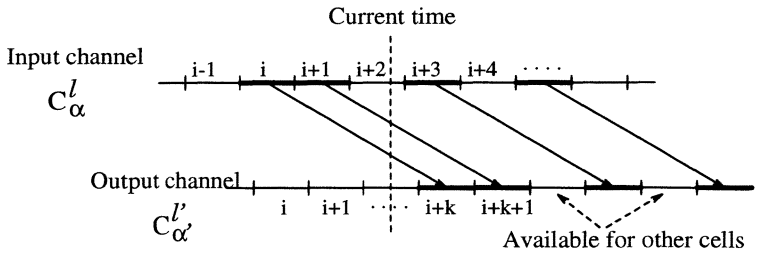

Figure 2: Delayed forwarding.

4. DELIVERY-ON-DEMAND OF UNCOMPRESSED DATA

Let f (cells) be the playback-frame size, F (cells) be the size of a multimedia file to be played back, R_c (cells/sec) be the consumption rate of the client's playback unit, and B (cells) be the buffer size available at the client site. In this section, given these parameters, we derive the minimum transfer rate, R_t^{min}, and maximum transfer rate, R_t^{max}, for uncompressed data. Any transfer rate R_t, satisfying $R_t^{min} \leq R_t \leq R_t^{max}$, can be used without causing buffer overflow or underflow. To avoid the discussion of trivial cases, we assume that $F > B$.

4.1 Range for Feasible Transfer Rate

Although the problem is to determine the range for feasible R_t for given B, F, and R_c, we first turn the problem around, and try to establish a lower bound on B for a given R_t. Since the interval time between two successive cells is the same on a given connection based on *Binary Preallocation* (see Section 3), to simplify analysis, we assume that the buffer fills with arriving cells as a linear function of time. (If the solution (i) suggested to solve Disadvantage 3 is used, this assumption does not hold and, therefore, the analysis must be slightly modified.)

Immediately after a frame is played back, the next frame should be available. This requires that $B \geq 2f$. We consider the two cases, $R_t < R_c$ and $R_t \geq R_c$, separately.

1. $R_t < R_c$:

To prevent buffer underflow, suppose we buffer as much data as possible before playback begins. Fig. 3 shows the number of cells in the buffer as a function of time. A played back frame is shown as being instantaneously consumed. Playback should start when there is only fR_t/R_c (cells) of free space left in the minimum-sized buffer. Thus, when the first frame has been played back, which takes f/R_c (sec), the buffer will be full except for one frame worth of free space, which has just been emptied. At this moment, the F–B remaining cells of the file are yet to be transmitted. The transmission of these cells must complete before the playback of the last frame of the file can start. We thus obtain

$$\frac{F-2f}{R_c} \geq \frac{F-B}{R_t}. \tag{4}$$

Let B_{min} denote the minimum possible value of B, which is a function of R_t/R_c. From

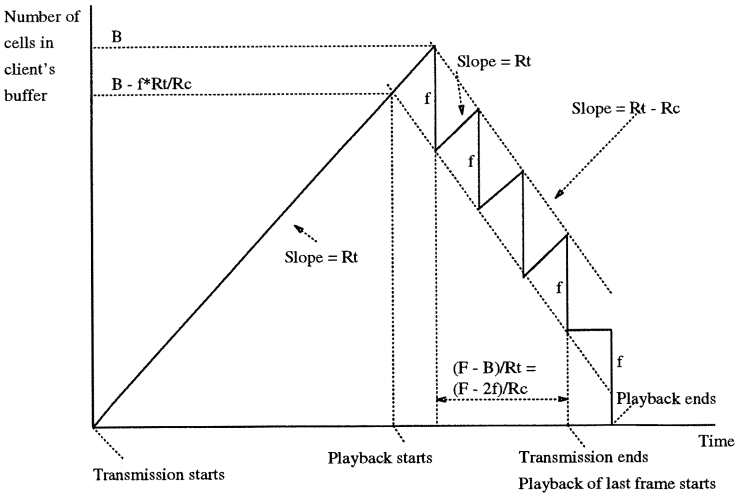

Figure 3: Buffered cells vs. time when $R_t < R_c$.

(4) we can derive

$$B_{min}/f = F/f - (F/f - 2)\frac{R_t}{R_c}. \tag{5}$$

2. $R_t \geq R_c$:

Since data arrive faster than they can be played back, we start playback as soon as a frame is available in the buffer. To analyze the buffer overflow problem, we consider two cases:

a) Transmission of the F–f remaining cells of the file takes longer than the playback of the first frame, i.e., $f/R_c < (F-f)/R_t$. (Fig. 4.)

b) Transmission of the remaining cells completes before the playback of the first frame finishes, i.e., $f/R_c \geq (F-f)/R_t$.

In the case b), we clearly need $B \geq F$, which is the uninteresting case that we ruled out earlier. Therefore, in the rest of this section, we consider only case a), assuming that $f/R_c < (F-f)/R_t$. In Fig. 4, transmission begins at time A, while playback starts at time S. The length of the line segment QS, denoted by $\overline{QS}$, is f, i.e., the amount of the prefetched data at the time playback starts, and $\overline{MD}$ is the amount of data in the buffer when transmission completes at time D. $\overline{OC}$ is the amount of data in the buffer at the time frame l (the last frame that finishes its playback no later than the time transmission completes) has just been played back. If $\overline{MD} > \overline{OC}$, as shown in Fig. 4, the maximum buffer space requirement occurs at time D. However, if $\overline{MD} < \overline{OC}$, the buffer contains more data at time C than at time D. Based on the above observations, we can derive

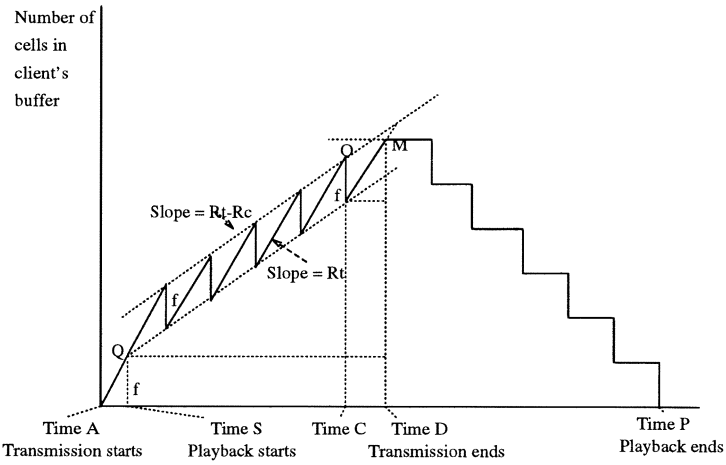

Figure 4: Buffered cells vs. time when $R_t > R_c$, $f/R_c < (F-f)/R_t$ and $f \leq F - f - \lfloor (F-f)R_c/fR_t \rfloor fR_t/R_c$.

the lower bound on B as $max\{\overline{MD}, \overline{OC}\}$, which can be broken down as in (6), which is more general than is necessary here: (We will need this general form in Section 5.)

$B \geq$ space for the prefetched data before playback starts
+space for data buffered between time S and time C
−space for data consumed between time S and time C
if the amount of data in frame l is less than the amount of data buffered after the playback of frame l finishes,
+space for data buffered after the playback of frame l finishes
else
+ space for frame l. (6)

Note that $\lfloor ((F-f)/R_t)/(f/R_c) \rfloor$ is the number of frames which have been played back before transmission ends. From (6), we obtain

$$B \geq f + \left\lfloor \frac{F-f}{R_t} \cdot \frac{R_c}{f} \right\rfloor f \left(\frac{R_t}{R_c} - 1 \right) + max \left\{ f, F - f - \left\lfloor \frac{F-f}{R_t} \cdot \frac{R_c}{f} \right\rfloor \frac{fR_t}{R_c} \right\}, \tag{7}$$

which can be rewritten as

$$\frac{B}{f} \geq 1 + \left\lfloor \left(\frac{F}{f} - 1 \right) \frac{R_c}{R_t} \right\rfloor \left(\frac{R_t}{R_c} - 1 \right) + max \left\{ 1, \frac{F}{f} - 1 - \left\lfloor \left(\frac{F}{f} - 1 \right) \frac{R_c}{R_t} \right\rfloor \frac{R_t}{R_c} \right\}. \tag{8}$$

(8) involves only "normalized" parameters, B/f, F/f and R_t/R_c. Both (5) and (8) yield $B_{min} = 2f$ at $R_t = R_c$. The interpretation of this is, of course, that when $R_t = R_c$, we need only two frames worth of buffer, regardless of the file size F; f for buffering the arriving data and f for playback.

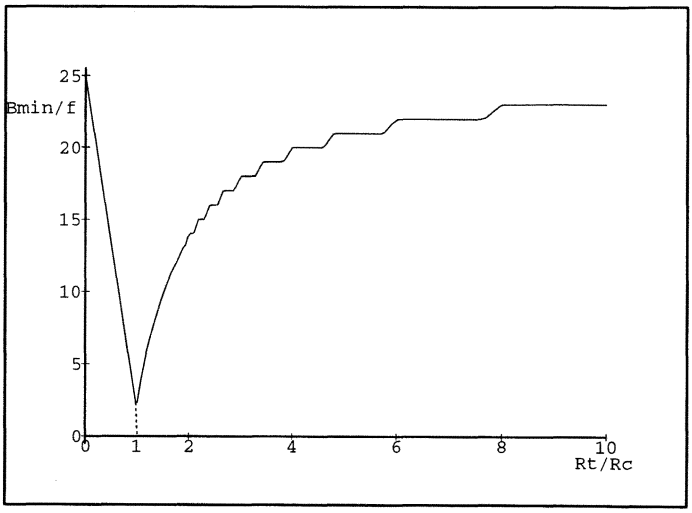

Figure 5: B_{min}/f vs. R_t/R_c. (*Maple* was used to draw this figure.)

Example 1: Based on (5) and (8), Fig. 5 illustrates the relationship between the (normalized) minimum required buffer size, B_{min}/f, and the (normalized) transfer rate, R_t/R_c, for $F/f = 25$. When $R_t/R_c \geq 1$, in some ranges the buffer space requirement remains constant while the ratio R_t/R_c increases. The flat segments correspond to the case illustrated in Fig. 4, i.e., where $f \leq F - f - \lfloor (F-f)R_c/fR_t \rfloor fR_t/R_c$, so that (8) yields $B_{min}/f = F/f + \lfloor (F/f-1)R_c/R_t \rfloor$. □

In practice, B/f (≥ 2) is a given constant, and we want to select a suitable transfer rate, R_t. In terms of a graph like Fig. 4, one can draw a horizontal line $B_{min}/f = b$, where bf is the available buffer size. In the cases of interest, this line intersects the curve at two points. The left and right intersections yield R_t^{min}/R_c and R_t^{max}/R_c, respectively. Any transfer rate, R_t such that $R_t^{min} \leq R_t \leq R_t^{max}$, can be used without causing buffer overflow or underflow.

4.2 Local Channel Allocation

In this subsection, we discuss how to select the transfer rate, R_t, within the range derived in the previous subsection. Recall that *Binary Preallocation* presented in Section 3 allocates a channel with bandwidth of the form $2^{-k}R(\ell)$ (cells/sec), in which k is an integer and $R(\ell)$ (cells/sec) is the total bandwidth of link ℓ. Therefore, if there is an integer k satisfying

$$R_t^{min} \leq 2^{-k}R(\ell) \leq R_t^{max}, \tag{9}$$

then a channel of bandwidth $2^{-k}R(\ell)$ can be allocated to this request, provided there is still enough unallocated bandwidth for the link. If there is one or more choices for k, any one can be chosen. However, if there is no integer k satisfying (9), then we should choose the smallest k such that $2^{-k}R(\ell) > R_t^{min}$, and set $R_t = 2^{-k}R(\ell)$. In this case, some slots

in the allocated connection will go unused. (See the discussion about Disadvantage 3 at the end of the previous section.)

Note that when we select a transfer rate during the channel establishment phase, we need only reserve a *bandwidth* of size $2^{-k}R(\ell)$, instead of a particular *channel*, C_α, of bandwidth $2^{-k}R(\ell)$, until it's been determined that a connection of bandwidth $2^{-k}R(\ell)$ is feasible.

4.3 Global Channel Allocation

As stated in Section 2, in order to establish a connection (or a global logical channel), we need to use a routing algorithm and an admission test at each node visited. Clearly, a fast routing algorithm is very important. However, we shall not specifically discuss this topic in this paper, referring the reader, instead, to references on known routing algorithms [1, 17].

We only comment on the possible resource contentions in the channel establishment phase. The network may be routing several requests simultaneously, each reserving some resources at the visited nodes. We abort a request (or divert it to another route) if required resources are not available at a node, so that no deadlock is possible.

5. DELIVERY-ON-DEMAND OF COMPRESSED DATA

In the case of compressed data, different playback-frames may consist of different numbers of cells, but the frame consumption rate, R_{fc} (frames/sec), is still constant. We assume that for every file, the minimum and maximum frame sizes, f_{min} (cells) and f_{max} (cells), respectively, are known, where $f_{max} > f_{min}$, in addition to the other parameters, i.e., F, B, and N (the number of cells in the file).

Thus, the *maximum consumption rate* is given by

$$R_c^{max} = f_{max} R_{fc} \quad \text{(cells/sec)},$$

and the *minimum consumption rate* is given by

$$R_c^{min} = f_{min} R_{fc} \quad \text{(cells/sec)}.$$

To avoid the discussion of trivial cases, we assume that $F > B$ and $N > 2$.

5.1 Buffer Underflow and Overflow

If $R_t \leq R_c^{min}$, then, clearly, there is no danger of buffer overflow. Similarly, if $R_t \geq R_c^{max}$, then there is no danger of buffer underflow. So, we need to address the buffer underflow problem only when $R_t < R_c^{max}$, and the buffer overflow problem only when $R_t > R_c^{min}$. Given a particular file, let S be the set of its frames, whose sizes range from f_{min} to f_{max}. Consider now the family of all files whose set of frames coincides with S. These files differ from each other in the sequence in which different frames appear in it. It turns out that a particular sequence gives rise to a more severe buffer under flow or overflow problem than others. For example, if the frames appear in the decreasing order in their sizes, then we need more buffer space than otherwise, in order to prevent buffer underflow. Intuitively, this is because, once playback is started, the buffer is more quickly emptied. Consider two files, $\mathcal{F}_1$ and $\mathcal{F}_2$, which are the same except that, in $\mathcal{F}_1$, frame a of size f_a is followed immediately by frame b of size f_b ($< f_a$), whereas, in $\mathcal{F}_2$, the order is reversed. Suppose playback starts for both files at the same time. It is not difficult to see that, if

Value of R_t	Problem encountered	Worst case
$R_t < R_c^{max}$	Buffer underflow	largest frames first
$R_t > R_c^{min}$	Buffer overflow	smallest frames first

Table 2: Problems for compressed data

buffer underflow occurs for $\mathcal{F}_2$ at the time the playback of frame b has just completed, then buffer underflow occurs for $\mathcal{F}_1$ as well at the time the playback of frame a has just completed. In other words, $\mathcal{F}_1$ is no better than $\mathcal{F}_2$, as far as preventing buffer underflow is concerned.

Above, we considered only those files which share the set S. Since, in our model, the only information we have about a file is its parameters, F, N, f_{max}, and f_{min}, we don't know what S is. The worst S, as far as preventing buffer underflow is concerned, consists of as many frames of size f_{max} as possible, and these frames all appear in the beginning of the file. Let N_{max} be the maximum number of frames of size f_{max}. We have $N_{max}f_{max}+(N-N_{max})f_{min} \leq F$, or $N_{max} = \lfloor (F-f_{min}N)/(f_{max}-f_{min}) \rfloor$. We can similarly show that the maximum number, N_{min}, of frames with size f_{min} can be expressed as $N_{min} = \lfloor (f_{max}N-F)/(f_{max}-f_{min}) \rfloor$. Table 2 summarizes the worst case for buffer space requirement.

5.2 Bounds on Transfer Rate

As in the case of uncompressed data, immediately after a frame is played back, the next frame should be available. This requires that $B \geq 2f_{max}$.
We divide the range of R_t into three parts:

(1) Region A: $R_t \leq R_c^{min}$ (only buffer underflow is possible).

(2) Region B: $R_c^{min} < R_t < R_c^{max}$ (both buffer underflow and overflow are possible).

(3) Region C: $R_t \geq R_c^{max}$ (only buffer overflow is possible).

Below, we present the minimum buffer space required, B_{min}, as a function of R_t. For the derivation of those results, the reader is referred to [18].

(1) Region A ($R_t \leq R_c^{min}$):

$$B_{min} = F - \frac{N-2}{R_{fc}} R_t. \tag{10}$$

Let $R_t^{min} = R_{fc}(F-B)/(N-2)$. If $R_t^{min} > R_c^{min}$ (i.e., $(F-B)/(N-2) > f_{min}$), there is no feasible transfer rate in Region A. Otherwise, we can select any R_t such that $R_t^{min} \leq R_t \leq R_c^{min}$. (See Region A in Fig. 6 for an example of (10).)

(2) Region B ($R_c^{min} < R_t < R_c^{max}$): Let

$$D_{pref} = f_{max}N_{max} - (f_{max}N_{max} - f_{max})\frac{R_t}{R_c^{max}} = f_{max}N_{max} - (N_{max}-1)\frac{R_t}{R_{fc}}, \tag{11}$$

and $\tilde{R}_t = N_{min}R_c^{min}/(N_{min}-N_{max}+2)$.

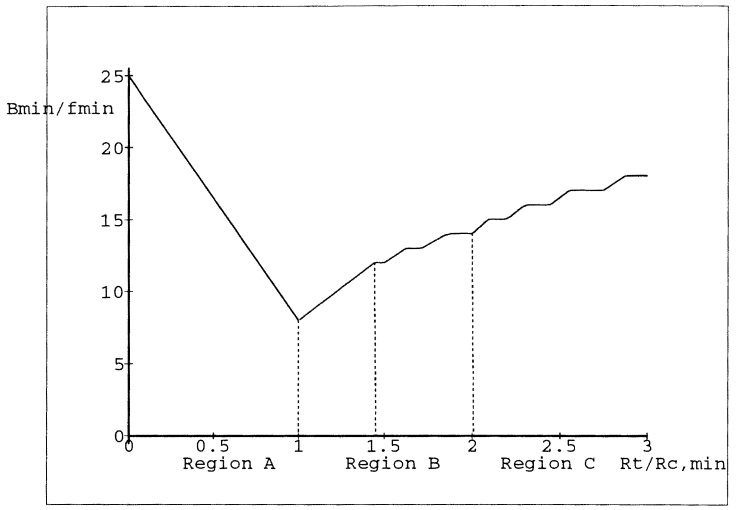

Figure 6: B_{min}/f_{min} vs. R_t/R_c^{min}. (*Maple* was used to draw this figure.)

a) When $N_{min}-N_{max}+2>0$ and $max\left\{\tilde{R}_t, R_c^{min}\right\} \le R_t \le R_c^{max}$:

$$\begin{aligned} B_{min} &= D_{pref} + \left\lfloor \frac{F - D_{pref}}{R_t} R_{fc} \right\rfloor \left(\frac{R_t}{R_{fc}} - f_{min} \right) \\ &\quad + max\left\{ f_{min}, F - D_{pref} - \left\lfloor \frac{F - D_{pref}}{R_t} R_{fc} \right\rfloor \frac{R_t}{R_{fc}} \right\}. \end{aligned} \tag{12}$$

b) When $N_{min}-N_{max}+2 \le 0$ and $R_c^{min} < R_t < R_c^{max}$, or when $N_{min}-N_{max}+2>0$ and $R_c^{min} \le R_t \le \min\left\{R_c^{max}, \tilde{R}_t\right\}$:

$$B_{min} = D_{pref} + N_{min}\left(\frac{R_t}{R_{fc}} - f_{min} \right) + \frac{R_t}{R_{fc}}. \tag{13}$$

(3) Region C ($R_t \ge R_c^{max}$):

a) When $R_t \ge max\left\{\bar{R}_t, R_c^{max}\right\}$:

$$\begin{aligned} B_{min} &= f_{max} + \left\lfloor \frac{F - f_{max}}{R_t} R_{fc} \right\rfloor \left(\frac{R_t}{R_{fc}} - f_{min} \right) \\ &\quad + max\left\{ f_{min}, F - f_{max} - \left\lfloor \frac{F - f_{max}}{R_t} R_{fc} \right\rfloor \frac{R_t}{R_{fc}} \right\}. \end{aligned} \tag{14}$$

b) When $R_c^{max} < R_t \le \bar{R}_t$: (If $\bar{R}_t \le R_c^{max}$, this case is impossible.)

$$B_{min} = f_{max} + N_{min}(f_{max} - f_{min}) + \left\lfloor \frac{F - f_{max}}{R_t} R_{fc} \right\rfloor \left(\frac{R_t}{R_{fc}} \right)$$
$$+max \left\{ f_{max}, F - f_{max} + N_{min} f_{min} - \left\lfloor \frac{F - f_{max}}{R_t} R_{fc} \right\rfloor \frac{R_t}{R_{fc}} \right\}. \quad (15)$$

Example 2: Fig. 6 illustrates the relationship between B_{min}/f_{min} and R_t/R_c^{min}. (Since R_c is variable for compressed data, we normalize R_t with respect to R_c^{min}.) The parameters used are $F = 250$ (cells), $N_{max} = 6$ and $N_{min} = 13$, $f_{max} = 20$ (cells), and $f_{min} = 10$ (cells). For these data, we have $\tilde{R}_t = 13/9$ and $\bar{R}_t = 23/13$. From Fig. 6, one can see that Region B ($R_c^{min} < R_t < R_t^{max}$) has two subregions, corresponding to case a) and case b), respectively. These two subregions meet when $R_t = \tilde{R}_t$. Since $\bar{R}_t < R_c^{max}$ for the specific values used in this example, only case a) is possible in Region C. If $\bar{R}_t > R_c^{max}$, there would be two subregions. We can also see that the minimum buffer space is required when $R_t = R_c^{min}$. For any fixed B_{min} larger than this minimum, a range of transfer rates, $[R_t^{min}, R_t^{max}]$, is feasible. □

6. CONCLUSION

We have presented a new channel allocation scheme for ATM-based networks, which has several important advantages over the currently known schemes. The most important advantage is that it will be able to satisfy the requirements of many real-time applications. We then discussed the delivery-on-demand of pre-recorded, continuous media data as a possible application of this scheme.

Although we assumed in our analysis that the buffer space available at the client site, B, was smaller than the file size, F, it is, of course, possible in practice that $B \geq F$. If so, then there is no restriction on the transfer rate, R_t. However, even in this case, our analysis is still useful in providing us with the minimum playback starting time, since minimizing the required buffer space also minimizes the playback starting time [6].

We are still in an early stage of investigation, and a lot of detail needs to be worked out. One such detail is the hardware implementation of the switches at network nodes. We are also currently investigating practical ways of allocating several subchannels of a link to a connection (see the discussion on Disadvantage 1 in Section 3).

References

[1] Bertsekas, D., and Gallager, R., *Data Networks,* Second Ed., Prentice-Hall, 1992.

[2] Clark J., "A Telecomputer," *Proc. SIGGRAPH '92*, Chicago, (July 1992), pp. 17-23.

[3] Decina, M., Open issues regarding the universal application of ATM for multiplexing and switching in B-ISDN, *Proc. ICC'91*, 1991, 1258-1264.

[4] Demers, A., "A high throughput bulk data transfer protocol," *Proc. SIGCOMM '89*, (Aug. 1989), pp. 154-167.

[5] Ferrari, D., "Client requirements for real-time communication services," *IEEE Comm. Magazine*, Vol. 28, No. 11 (Nov. 1990), pp. 65-72.

[6] Gemmell, J., "Principle of delay-sensitive multimedia data storage and retrieval," *ACM Trans. Information Systems*, Vol. 10, No. 1 (Jan. 1992), pp. 51-90.

[7] Golestani, S.J., Congestion-free communication in high-speed packet networks, *IEEE Trans. Comm.*, Vol. Comm-39, No. 12 (Dec. 1991), pp. 1802-1812.

[8] Kopetz, H. and Ochsenreiter, W., Clock synchronization in distributed real time, *IEEE Trans. Computers,* C-36, No. 8 (Aug. 1987), 933-940.

[9] Kurose, J., Open issues and challenges in providing quality of service guarantees in high-speed networks, *Computer Communication Review*, Vol. 23, No. 1 (Jan. 1993), 6-15.

[10] Li, C-S., Ofek, Y., Segall, A., and Shoraby, K., Pseudo-isochronous cell switching in ATM networks, *Proc. INFOCOM'94*, June 1994, pp.428-437.

[11] Minzner, S.E., "Broadband ISDN and asynchronous transfer mode (ATM)," *IEEE Comm. Magazine* Vol. 27 (1989), pp. 17-24.

[12] Ofek, Y., Generating a fault tolerant global clock using high-speed control signals for the MetaNet architecture, *IEEE Trans. Comm.* Vol 42, No. 5 (May 1994), pp. 2179-2188.

[13] Peha, J.M., and Tobagi, F.A., Evaluating scheduling algorithms for traffic with heterogeneous performance objective, *GLOBECOM'90*, 1990, pp. 21-27.

[14] Press, L., "The Internet and interactive television," *Comm. ACM,* Vol. 36, No. 12 (Dec. 1993), pp. 19-23.

[15] Ramanthan, S., and Rangan P.V., "Feedback techniques for intra-media continuity and inter-media synchronization in distributed multimedia systems," *Computer J.*, Vol. 36, No.1 (Jan. 1993), pp. 19-31.

[16] Rathgeb, E.P., "Comparison of policing mechanisms for ATM networks," *Proc. 3rd RACE Workshop*, (Oct. 1989), pp. 211-224.

[17] Stallings, W., *Data and Computer Communications,* Third Ed., Macmillan, 1991.

[18] Ting, J., Kameda, T., and Fracchia, D., Delivery-on-demand of continuous media data, *Tech. Rept.* LCCR/CSS 94-04, School of Computing Science, Simon Fraser University, 1994.

[19] Woodruff, G.M., and Kostpaiboon, R., "Multimedia traffic management principles for guaranteed ATM network performance," *IEEE J. Selected Areas in Comm.*, Vol. 8, No. 3 (Apr. 1990), pp. 437-446.

[20] Zhang, L., "Virtual Clock: A new traffic control algorithm for packet switching networks," *ACM Trans. Computer Systems*, Vol. 9, No. 2 (May 1991), pp 101-124.