REAL-TIME COMMUNICATION IN A PACKET-SWITCHING NETWORK

Domenico Ferrari and Dinesh C. Verma
Computer Science Division
Department of Electrical Engineering and Computer Sciences
University of California
and International Computer Science Institute
Berkeley, California, U.S.A.

Abstract

Broadband integrated networks will have to offer *real-time communication services*; that is, they will have to transport information with performance guarantees. A real-time service is defined in this paper as one capable of providing connections with performance guarantees (the *real-time channels*) on demand and ensuring that the guarantees are always satisfied. These guarantees can be of various types, e.g., a lower bound on the throughput of a connection, an upper bound on delays experienced by a packet on a channel, or an upper bound on the packet loss rate.

In this paper, we study the feasibility of providing real-time services on a packet-switched store-and-forward wide-area network with general topology. We describe a scheme for the establishment of channels with deterministic or statistical delay bounds and bounded loss rates, and present the results of the simulation experiments we ran to evaluate it. The results are encouraging: our approach satisfies the guarantees even in worst-case situations, uses the network's resources to a fair extent, and efficiently handles channels with a variety of offered load and burstiness characteristics. Also, the packet transmission overhead is quite low, and the channel establishment overhead is small enough to be acceptable in most cases.

This research has been supported in part by AT&T Bell Laboratories, Hitachi, Ltd., the University of California under a MICRO grant, the National Science Foundation under Grant No. CDA-8722788, and the International Computer Science Institute. The views and conclusions in this document are those of the authors and should not be interpreted as representing official policies, either expressed or implied, of any of the sponsoring organizations.

1. Introduction

Broadband integrated networks will have to offer *real-time communication services*, i.e., to transport information with performance guarantees, primarily to satisfy the requirements of digital audio and digital video [Lein89]. The specific performance guarantees that will be needed will vary with the type of traffic (video, audio, or their various combinations with each other and with other information media) as well as with the application [Ferr90]. Once such services become available, they are likely to be used even for types of traffic that do not include video or audio, for example, alarm conditions in distributed real-time systems, urgent remote database queries, or file transfers requiring lower bounds on throughput. Thus, real-time communication services are going to be essential in the integrated networks of the future, along with the traditional, non-real-time ones such as datagram and virtual circuit services of various kinds.

Three basic approaches can be used to provide real-time communication: circuit switching, packet switching, and hybrid switching. A *circuit-switching* network will allocate its resources continuously and exclusively to each connection for the entire duration of the connection [Harr80]. With *packet switching*, resource allocation is neither continuous nor exclusive; that is, resources are shared among various conversations on a packet-by-packet basis, and the only privileges certain conversations may receive consist of a higher priority (preemptive, but with a non-preemptable granule, i.e., the packet) in the use of shared resources. In a *hybrid-switching* network, the two previously described approaches coexist: certain conversations (typically, those needing performance guarantees) are carried by the circuit-switching portions, others by the packet-switching portion of the network.

Both circuit switching and hybrid switching result in poor resource utilization when bursty traffic requires performance guarantees. It is therefore useful to investigate whether packet switching can be used to provide performance guarantees, and, if so, whether and under what conditions a packet-switching solution is more convenient than the other two.

This paper describes a method for guaranteeing delays and loss rates in a packet-switching wide-area network, and presents an evaluation of some of its most important characteristics. We believe that the method can be used in all connection-oriented packet-switching environments, including ATM-style fast packet switching. However, we present it in the context of a contemporary connection-oriented packet-switching wide-area network, and we evaluate it by simulation in the same context. We state the problem precisely in Section 2, sketch our approach to it in Section 3, and illustrate the heart of our solution, i.e., the establishment algorithm, in Section 4. Section 5 describes the simulations we ran to evaluate the scheme, and discusses the results we obtained. Finally, conclusions are drawn in Section 6.

Ideas similar to that of a real-time channel can be found in [CoYa88], [Zhan89] and [OhON88].

2. Problem Statement

In its full generality, the network we consider in this study has an arbitrary topology, and may consist of several networks of various types. In it, a message can go from a source host to a destination host passing through a number of intermediate store-and-forward nodes, some of which may be gateways connecting different networks. Some of the links between adjacent store-and-forward nodes may actually be non-store-and-forward networks, provided their total delay can be bounded, for example, very high-speed circuit-switched trunks, or FDDI rings[SeJo87]. Note that some networks governed by contention-based protocols (e.g., Ethernets) do not satisfy this property.

We assume that real-time communication will be based on simplex fixed-route connections to be called *real-time channels* or simply *channels*, whose routes will be chosen at the establishment time. The connection-oriented approach unfortunately requires explicit establishment, which may be slow. Our solution tries to reduce the establishment time by limiting the connection establishment procedure to a single source-destination round trip and by speeding up as much as possible the establishment algorithms. Our approach thus assumes that the life of a connection will be much longer than the connection establishment time. For those cases in which the life-time of a connection is so short as to make an explicit connection establishment message inefficient, one could use a fast establishment scheme [DaVe89] or multiplex connections on pre-existing relatively permanenet real-time channels.

In order to provide real-time service, we require that the clients declare their traffic characteristics and performance requirements at the time of channel establishment. There are various meaningful ways in which the offered load and the performance bounds of a channel can be specified. For our first study in this area, we have chosen the following parameters:

- for the offered load
  - the minimum packet interarrival time on the channel, x_{min},
  - the minimum value x_{ave} of the average packet interarrival time over an interval of duration I,
  - the maximum packet size s_{max}, and
  - the maximum service time t in the node for the channel's packets;
- for the performance bounds
  - the source-to-destination delay bound (or bounds) for the channel's packets, and
  - the maximum packet loss probability.

Two types of channels are considered in this paper:

- *deterministic*: the bound D for packet delays is an absolute one; this is necessary in hard real-time applications;
- *statistical*: the bound is expressed in statistical terms; for instance, the probability that the delay of a packet is smaller than the given bound D must be greater than a given value Z.

The channel's loss requirements are specified by means of the maximum loss probability W. Of all the packets that are shipped on a real-time channel, at least $100W$% must be delivered correctly. If the channel is statistical, at least $100Z$% of those delivered correctly must be delivered on time. Thus, the packets that are delivered correctly and on time will be at least $100WZ$% of those that were shipped by the sender. Some clients may be more interested in WZ than in the individual values of these two bounds, and may want to leave the decision about what these values should be to the real-time server [Ferr90].

Channel i will be characterized by its own values of the offered load and performance bounds parameters; we will add the subscript i to the symbols of those parameters (e.g., we will write $x_{min,i}$, D_i, and so on) whenever necessary to avoid ambiguities.

We assume that the network is characterized by a known error rate, and that the given value of W will be adjusted accordingly to obtain the lower bound for the probability of buffer overrun. For the sake of simplicity, we shall use W in the sequel to denote not the correct-delivery probability bound, but one of its two factors, namely, the lower bound on the probability that no buffer will overflow, thereby causing packet loss. We also assume that the traffic on the channel to be established is statistically independent of the remaining channels sharing totally or partially their route.

3. A Solution

Our approach consists of a scheme (to be described in Sections 3.3 and 4) for real-time channel establishment. The scheme works only in conjunction with specific scheduling and flow control policies. These policies are briefly described in Sections 3.1 and 3.2, respectively.

3.1. Scheduling

We assume that scheduling in the hosts and in the nodes will be deadline-based. Each packet in the node is given a deadline (see Section 3.3 for details), which is the time by which it is to be serviced. We adopt a modification of the EDD (Earliest Due Date) [LiLa73] policy in which priority is given to deterministic channels over statistical ones in the case of a conflict. All of the other tasks of a host or node, including sending, forwarding, and receiving datagrams, have a lower priority; local tasks are preemptable by real-time packets.

In each node, the scheduler maintains at least three queues: one for deterministic packets, one for statistical packets, and the third for all other types of packets and all local tasks. The third queue can in turn be replaced by multiple queues, managed by a variety of policies, but we are not concerned with these aspects of node scheduling here.

A packet arriving at the deterministic queue is to be ‘‘aligned’’, that is, its deadline must be appropriately reduced if its service time would overlap with that of another deterministic packet in case they both happened to be shipped at their latest possible times. [1] The ordering of the first two queues is by increasing deadlines.

When the node must choose the next action to be undertaken, the scheduler compares the end time (i.e., the deadline) of the packet at the head of the statistical queue with the beginning time (i.e., the deadline minus the service time) of the head packet in the deterministic queue: if the latter is lower than the former, the deterministic packet is scheduled to be shipped immediately; otherwise, the statistical packet is shipped. Tasks and packets in the third queue are processed only when the first two queues are empty.

3.2. Rate control

At channel establishment time, the receiver checks (using the method described in Section 4) whether it will be able to accept packets at the rate declared by the sender. However, a malicious user could circumvent any rate-based flow control mechanism and from a single-user machine send packets into the network at a much higher rate than the declared maximum value, $1/x_{min}$, or maximum average value, $1/x_{ave}$. The same effect might be caused by a failure in the

[1] An example of a situation requiring alignment is when two unit-sized packets arrive at time 0 with delay bound of 4 units each; due to alignment one of them is given a deadline of 4 while the other is given a reduced deadline of 3.

sending host. Such malicious or faulty behavior can prevent the satisfaction of the delay bounds guaranteed to other clients of the real-time service, thereby damaging the clients and destroying the credibility of the service.

Our solution to the problem consists of providing distributed rate control by increasing the deadlines of the ‘‘offending’’ packets, so that they will be delayed in heavily loaded nodes, where they would otherwise seriously interfere with the operation of other channels. The deadline assigned to an offeding packet would equal the deadline that packet would have if it had obeyed the $x_{\min}$ and x_{ave} constraints declared at the connection establishment time. If the client continues to send data in excess of the promised rate, the number of dropped packets is no longer constrained by the promised loss bound W. A very simple algorithm for distributed rate control is presented in detail in [Ferr89].

One important consequence of this scheme for the algorithms in Section 4 is that we can assume that each channel satisfies its x_{min} and x_{ave} constraints at every node it traverses, at least for the purpose of guaranteeing delay bounds. The buffer space allocation, however, must make sure that no packets of a law-abiding client are dropped, due to network load fluctuations, beyond the promised bound.

3.3. The channel establishment procedure

The channel establishment mechanism we propose for the type of network described in Section 2 may be built on top of any procedure that can be used to set up connections. Besides trying to establish a connection, the mechanism will perform several tests and tentatively reserve resources in each node visited by the establishment request message. In order to make channel establishment fast, we impose on our procedure the restriction that it require only one round trip. Thus, the destination host is the last point along the path where the acceptance/rejection decision for a channel request can be made. When a node is revisited by an establishment message during this message’s return trip, the resources previously reserved there must be committed or released; hence a final, irreversible decision must have already been made.

The tests to be done in each node are concerned with the availability of sufficient bandwidth in the links, as well as processing power and buffer space in the node; that is, with determining whether the new channel can go through the node without jeopardizing the performance guarantees given to the already established channels passing through the same node.

If any test fails at a node, the channel cannot be established along that route; the message will be sent back, either to the sender (which may then decide to wait or try another output link) or to an intermediate node that can try sending the message towards the destination along another path. When an unsuccessful establishment request message revisits a node on its way back to the source, it frees all the resources that were tentatively reserved there during its forward trip.

If all the tests for channel i succeed at all nodes and at the destination host, this host subdivides the delay bound D_i, the probability Z_i of not exceeding the bound if the channel being established is statistical, and the probability W_i of not losing a packet due to buffer overruns, among the nodes traversed by the channel, after accounting for the link delays and error rates along the route. Let $d_{i,n}$ be the delay bound assigned to the channel in a node n, and $z_{i,n}$ the probability that $d_{i,n}$ will not be exceeded in that node. A packet traveling on that channel and arriving at that node at time t will usually [2] be assigned a node deadline equal to $t + d_{i,n}$.

[2] There may be exceptions due to alignment (see Section 3.1) or distributed rate control (see Section 3.2).

If all tests succeed, a reply message is sent back to the source host along the channel's route; this message notifies each node about the bounds that have been assigned to it for the new channel, and commits the necessary portions of the reserved resources. When the reply message reaches the source host, this host learns that the requested channel has been set up, and can start using it.

The next section describes in some detail the algorithms for channel establishment to be executed in each node along the new channel's route and in the destination host.

4. The Algorithms

When a node receives an establishment request message, it performs some or all of the following tests:

(a) the *deterministic test*, required when the channel to be established is deterministic, and involving the deterministic channels already passing through the node;

(b) the *statistical test*, to be performed for the establishment of both statistical and deterministic channels if at least one statistical channel is already passing through the node or is to be set up; this test involves all deterministic and statistical channels passing through the node;

(c) the *delay bound test*, which is needed in all cases; if successful, it is to be followed by the computation of the minimum feasible delay bound for the new channel in the node;

(d) the *buffer space test*, which is needed in all cases; if successful, it is to be followed by the computation of the buffer space to be reserved for the new channel and the probability that there will not be buffer overruns for the new channel.

If the request passes the tests, then the node sends the establishment message on to the next node, after having added to it some of the parameter values resulting from the tests listed above. The destination host performs the final tests, and then, if the request passes these too, subdivides D (Z) and W among the nodes on the channel's path. We now briefly describe each one of the tests as well as the destination host algorithms.

4.1. The deterministic test

The deterministic test consists of verifying that enough processing power is available in the node to accommodate the additional deterministic channel without impairing the guarantees given to the others. Since we are dealing with deterministic bounds, this condition must be satisfied even in the worst possible case, i.e., even when all deterministic channels are sending packets into the node at their maximum rates. The maximum utilization of a node n by channel j, whose packets have a maximum service time in the node equal to $t_{j,n}$, is $t_{j,n}/x_{min,j}$. The condition to be tested is:

$$\sum_j (t_{j,n}/x_{min,j}) < 1, \tag{1}$$

where the sum extends to all deterministic channels passing through node n, including the one to be established (channel i).

4.2. The statistical test

The statistical test has two goals:

(i) to determine whether for each statistical channel j passing through node n the probability of a delay higher than the bound $d_{j,n}$ is below its maximum tolerable value, 1 - $z_{j,n}$;

(ii) to provide the destination host with the information necessary to compute $z_{i,n}$ for the new channel if this is statistical.

Both these goals can be attained by computing the *probability of deadline overflow* $P_{do,n}$. A packet may be delayed beyond its delay bound in a node because of two reasons: a temporary saturation of the node's processing or transmission capacity (*node saturation*), and impossible scheduling constraints (*scheduler saturation*).[3] While the latter is avoided, as discussed in Section 4.3, the statistical test deals with the former.

Let at least one of the channels be statistical. To compute $P_{do,n}$, we start by listing all the *overflow combinations*. An overflow combination is a set of channels that, when simultaneously active for a sufficiently long time, may cause packets to miss their deadlines. In other words, an overflow combination is one for which inequality (1) above, with the sum extended to all active channels, is not satisfied. In making this statement, we assume a worst-case situation, i.e., one in which each active channel carries packets at its maximum rate. If the input load were to persist indefinitely, the length of the queue of packets in the node would grow without bounds. Let $P_{do,n}$ be the probability that such an overflow combination is present; this probability can be computed by a simple method described in [FeVe90]. We can then check whether the following inequalities are satisfied:

$$P_{do,n} \leq 1 - z_{j,n}, \tag{2}$$

for all statistical channels j existing in the node.

If test (2) is successful for all statistical channels j in the node, the value of $P_{do,n}$ is sent to the destination host, which will try to assign to channel i a value of $z_{i,n}$ that satisfies inequality (2).

4.3. The delay bound test

The delay bound test determines the minimum delay bound to be assigned to the channel being established so that scheduler saturation can be avoided in the node. We try to eliminate this type of saturation primarily because of the complexity of analyzing it in the general case.

In the statistical test (see Section 4.2), we took into account node saturation, which delays packets beyond their deadlines whether or not scheduler saturation is present. Thus, in those situations in which the node is saturated, whether we have scheduler saturation or not is immaterial. However, scheduler saturation could occur also in cases in which the node is not saturated. To determine whether this will indeed happen, and to prevent it from happening, we must consider the worst possible combination, i.e., one that maximizes the likelihood of scheduler saturation without saturating the node. If the combination corresponding to all

---

[3] A simple example of scheduler saturation is that of two channels with respective node service times 3 and 4 units, and respective node delay bounds 5 and 6 units; if two packets from the two channels arrive simultaneously at the node, there is no way to schedule them so that both meet their deadlines.

channels through the node being active saturates the node, we will have to consider for the delay bound test a combination in which some of the channels are not active. Details about how such a combination can be arrived at are given in [Ferr89].

Let the cardinality of this selected combination of channels be K. Let us denote by T the largest of the service times of packets that may traverse the node. With no loss of generality, we assume that the selected channels are numbered 1 through K, according to the order in which their packets would be scheduled by the algorithm described in Section 3.1 if they arrived all at the same instant: channel 1 will be the one whose packet would be shipped first, channel K the one whose packet would be shipped last. Let us also assume for simplicity of proof that

$$x_{min,i} \geq \sum_{j=1}^{K} t_{j,n}, \quad (i=1,...K). \tag{3}$$

The following theorem is the basis for the delay bound test.

Theorem: Scheduler saturation is impossible if and only if

$$d_{i,n} \geq \sum_{j=1}^{i} t_{j,n} + T, \quad (i=1,...K). \tag{4}$$

A proof of this theorem can be found in [FeVe90], where we briefly discuss also the cases in which inequalities (3) do not hold. [4] [Ferr89] contains a few examples of delay bound computations.

The delay bound test consists of verifying that inequality (4) is satisfied for all values of i (i.e., for all the selected channels), after the channel to be created has been added to the K already established ones. While testing the delay bounds of the established channels, if the delay bounds pass the test it is easy to determine the lower bound $d^{l}_{i,n}$ of $d_{i,n}$ such that scheduler saturation will be impossible in the node if we choose $d_{i,n} \geq d^{l}_{i,n}$. The value of $d^{l}_{i,n}$ will then be sent to the destination host in the establishment message.

4.4. The buffer space test

The buffer space test determines whether there is sufficient space to accommodate the new channel, and how much of the existing space should be reserved for it. It can be shown [FeVe90b] that the amount of buffer space $b_{i,n}$ required by a deterministic channel i at node n to insure that the loss probability is less that $1 - w_{i,n}$ is given by

$$b_{i,n} = s_{max,i} \left\lceil \frac{d_{i,n}}{x_{min,i}} \, w_{i,n} \right\rceil . \tag{5}$$

Similarly, the buffer space required for a statistical channel i at node n when node n has an overflow probability $P_{do,n}$ is given by

$$b_{i,n} = s_{max,i} \left\lceil \frac{\left\lceil d_{i,n} + I\, P_{do} (\sum_{j} \frac{t_{j,n}}{x_{min,j}} - 1) \right\rceil}{x_{min,i}} w_{i,n} \right\rceil \tag{6}$$

[4] For ease of presentation, we have glossed over a few technicalities which would make the theorem much stronger. The interested reader is referred to [Ferr89] for all the details.

If at least one statistical channel exists at the node, the buffer space test consists of determining whether the space allocated to each existing statistical channel is adequate to meet its requirements, even after accepting the new channel. Otherwise, the space allocated to the channel is increased.

The next step is to compute the amount of buffer space to be reserved for the new channel. Since at the time of the forward trip, neither $d_{i,n}$ nor $w_{i,n}$ are known, the node assumes worst case upper bounds for them. The simplest way is to bound $d_{i,n}$ by D_i and $w_{i,n}$ by 1. If the space required by this (admittedly crude) guess is not available, the value of $w_{i,n}$ is reduced appropriately to reflect this fact. The values of $b_{i,n}$ and $w_{i,n}$ are sent to the destination host in the establishment message.

4.5. The destination host tests and algorithms

The destination host, if and when it receives the establishment request message, has to determine whether the total value of Z_i for the new channel i can be factored into node contributions $z_{i,n}$ (n = 1, 2,...N):

$$Z_i = \prod_{n=1}^{N} z_{i,n}. \tag{7}$$

This question can be answered by checking whether

$$Z_i \le \prod_{n=1}^{N} (1 - P_{do,n}). \tag{8}$$

This may be called the *Z test*. If (8) does not hold, then the request is rejected. Note that (7) implies the assumption that, once a packet is delayed beyond its deadline in a node, it will not be able to satisfy the channel's overall delay bound. Rejecting a request if (8) does not hold is a policy based on worst-case considerations, which guarantee that the desired results will be achieved in all possible circumstances. Of course, there is a cost associated with worst-case design: these and other conservative decisions reduce the maximum number of channels that can be established in a given network with respect to that which the same network could support if the design of the real-time service were less conservative.

The test concerned with the delay bound (the *D test*) consists of verifying that, for channel i with total delay bound D_i to be established, we have

$$D_i \ \ge \ \sum_{n=1}^{N} d^l_{i,n}. \tag{9}$$

Two simple expressions for assigning the $d_{i,n}$'s and $z_{i,n}$'s to the nodes along the channel's path are

$$d_{i,n} = \frac{1}{N}\left[D_i - \sum_{m=1}^{N} d^l_{i,m} \right] + d^l_{i,n}, \tag{10}$$

$$z_{i,n} = \left[Z_i / \prod_{m=1}^{N} (1 - P_{do,m}) \right]^{1/N} (1 - P_{do,n}). \tag{11}$$

On the basis of the offered values of the various performance bounds (i.e. ones which came in the establishment message), the destination host can invert equation (6) to obtain the value of $P_{do,n}$ at each of the nodes along the path. It can then use the final values of $d_{i,n}$ and $z_{i,n}$ (computed from equation (10) and (11)) to improve upon the offered value of $w_{i,n}$. This value will certainly be better than the one computed by the intermediate node in Section 4.4, where it used upper bounds for d_i and z_i. The destination host then checks that

$$W_i \geq \prod_{n=1}^{N} w_{i,n}. \tag{12}$$

This may be called the *W test*. The assumption made in this test, that a packet lost at any node is lost forever, is obviously correct. If (12) does hold (otherwise the channel is rejected), then the final values of $w_{i,n}$ can be computed by a simple formula analogous to equation (11).

The destination finally computes the actual buffer space required at each node in the path for the channel to be established using equation (5) or (6) as appropriate. Note that this computation can only decrease the amount of buffer space offered during the first pass in the buffer space test (see Section 4.4) and never increase it.

5. The Simulations

In this section, we attempt to provide simulation-based answers to the following questions which the reader may have asked about the scheme.

- Is the scheme correct? Can we discover cases when the loss or delay guarantees can not be met using this scheme?
- How much time is required to perform the various tests?
- Providing the performance guarantees involves a number of pessimistic assumptions. What are the effects of these assumptions?

We obtained answers to these questions by means of a simulator written in CSIM [Schw89]. In this section, we describe some of the simulation results.

Future multimedia networks will have a wide variety of traffic types. To capture some of this diversity, we consider four types of channels. Table 1 shows the types of channels used in the simulation experiments. Type I channels tax the network resources heavily, while type IV channels are very light.

Table 1. Channel types

Channel Type	t (time units)	$x_{\min}$ (time units)	x_{ave} (time units)
I	4	10	60
II	4	40	120
III	1	10	60
IV	1	40	120

The simulation was run in two phases. In the first phase, we tried to establish as many channels as we could. After the establishment of all the channels, in the second phase, packet traffic on all of the channels was simulated. We experimented with various kinds of traffic and will be presenting the results of the worst traffic pattern that we tried. Thus, the results presented in this section deal with the worst possible situations, i.e., ones that result in the maximum possible losses and delays.

The results of one first phase simulation on an arbitrary network are shown in Figure 1. The number reported near each node is the expected utilization of the node by the established

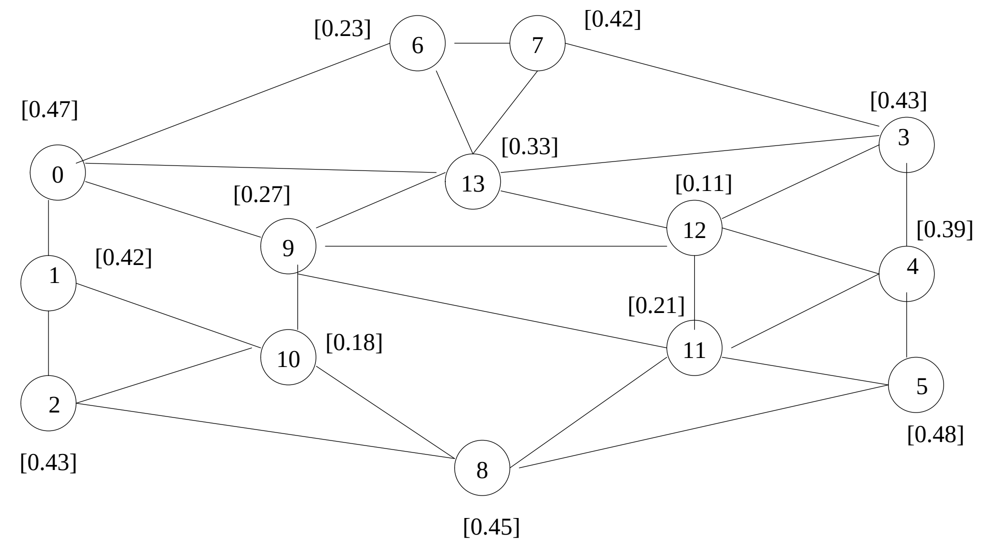

Fig 1. Saturated Network

Figure 1. A network and the utilization of different nodes in the network as a result of our algorithms. The number reported near each node is the expected utilization of the node by the channels traversing it. The delay and buffer distributions of some channels at node 0 have been plotted later on.

channels. We selected this configuration, containing about 50 (24 deterministic and 26 statistical) successfully established channels for second phase simulations.

In the second phase, we experimented with a number of different arrival patterns. The worst pattern, i.e., the one which resulted in the maximum delays and losses, appeared to be a pattern in which packets arrived either at an interval of $x_{\min}$ or at sufficiently large intervals that brought the average in a period of duration I to x_{ave}. Thus, the arrival pattern consisted of a burst of $\lceil I/x_{ave} \rceil$ packets followed by a period of silence. In order to satisfy the independence assumptions involved in the computation of the overflow probability, all the traffic patterns were given a random offset in every interval of length I, which staggered them relative to one another.

We also experimented with other arrival patterns, for example, one in which the burst sizes were distributed geometrically, one in which arrivals occurred smoothly at the average rate and one in which the inter-arrival time was uniformly distributed between the minimum and an upper limit so as to achieve the specified average value. However, we will only describe the results of the worst arrival pattern, i.e., one in which the arrivals consisted of the maximum possible burst size followed by a long period of silence.

Figure 2 shows the delay distribution of a statistical and a deterministic channel at node 0 (which had a utilization of 47%). The two channels had identical traffic characteristics. The statistical channel had a delay bound of 30 units at node 0 with *z* (probability of meeting this delay) equal to 0.92 and *w* (probability that a packet will not be lost) equal to 0.96. The deterministic channel had the same bound on the loss probability, but its delay bound was slightly higher (34 time units).

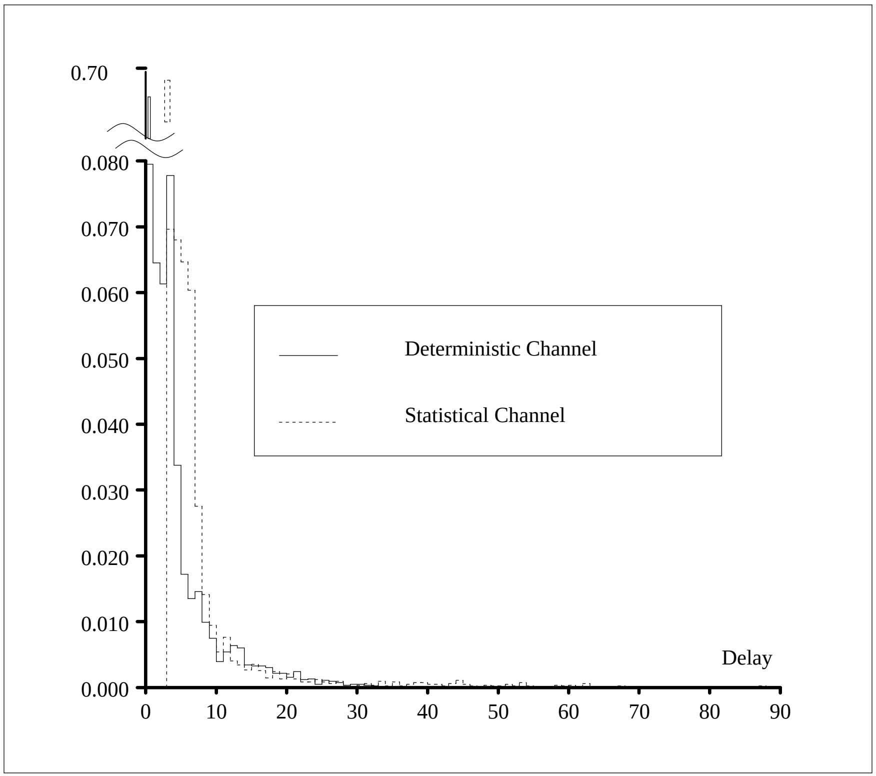

Figure 2. Delay distributions on real-time channels. A large fraction of the packets had a very low delay (which is why there is a break at the top left corner of the histograms). The delay bound of the statistical channel was 30 units, and about 2% of the packets failed to meet the required delay bound. The maximum delay, however, could be substantially higher than the assigned bound, e.g., in this figure the maximum delay is 88 units, about three times the bound of 30 units. All packets on the deterministic channel met the delay bound of 34 units. Both channels are of type III (see Table 1).

Figure 2 shows that a packet is likely to have a very low delay although for some statistical packets the delay may be as high as three times its assigned bound. In the simulations, 98.6% of the packets on the statistical channel shown in Figure 2 were transmitted within the required bound of 30 units.

Figure 3 shows a histogram of the number of packets present in the queue (or buffer occupancy) as seen by packets arriving at node 0 on the same channels to which Figure 2 refers. Most arrivals find no packet from the same channel in the queue. The equations in Section 4.4 predict that space for six packets will be sufficient to ensure a loss probability of less than 4% for the statistical channel and that four buffers will be sufficient for the deterministic channel. In the actual simulations, less than 0.04% of the packets were lost due to buffer overruns, for the statistical channel.

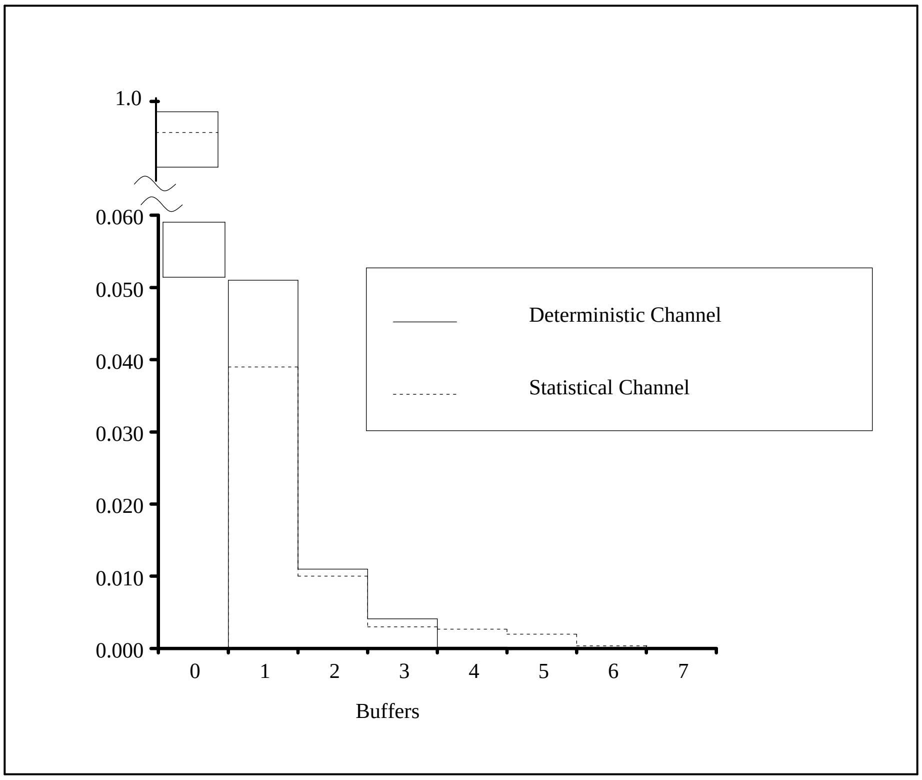

Figure 3. The buffer occupancy distribution for the deterministic and statistical channels shown in Figure 2. The figure shows the histogram of the number of packets on the same channel present in the channel's queue at node 0 as seen by arriving packets. Six buffers were allocated to the statistical channel, hence the arrivals which found six packets already in the queue were dropped. The number of packets lost this way, however, was only 0.04%, well within the 4% bounds that was guaranteed. A large fraction of the packets find no buffers occupied, as shown by the break in the histogram at the top left corner.

Thus, Figures 2 and 3 illustrate that the aforementioned formulas are amply sufficient to meet the required performance bounds for both deterministic and statistical channels.

5.1. Overhead

The overheads of the buffer space test and the statistical test are linear in the number of channels already established at a node. Similarly, the overheads of the destination algorithms are linear in the number of the nodes along the path. The deterministic test, however, can be done in constant time. To give a quantitative estimate, with between 8 and 18 channels established at various nodes, the mean time to perform the buffer space tests on a VAX-8600 was about 1.0

millisecond at every node. The run time overhead for buffers consisted of a simple increment and comparison at packet arrival and a simple decrement at packet departure, which did not take any significant amount of time.

In our simulated environment the other tests were also quite efficient. The deterministic test took about 30 microseconds and the statistical one about 0.2 milliseconds to execute. Book keeping overhead during forward and reverse trips was around 2 milliseconds per node, excluding the time to calculate the overflow probability P_{do}. The destination spent about 0.61 milliseconds trying to allocate performance bounds to different nodes along the route of the channel to be established.

The calculation of P_{do} is dependent on the desired degree of accuracy we want our approximate calculations to achieve. In the simulations, we opted for an accuracy to the third decimal place, which required about 10 milliseconds for a single computation. If we had opted for an accuracy of only two decimal places, the time taken would have been around 1 milliseconds only.

The scheduling overhead per packet in our simulations was about 60 μs, and rate control had an overhead of 40 μs. While it would be hard to claim that our simulations exactly reflect the situation in the real world, these results show that the overhead is low enough to be acceptable in many situations. Of course, a hardware implementations of the scheduling scheme would be much faster.

A more comprehensive simulation-based study of the scheme, including its behavior under different levels of burstiness can be found in [FeVe90] and [FeVe90b]. Because of space limitations, we will not be presenting them in this paper.

6. Conclusions

This study was undertaken to determine the feasibility of offering real-time services in packet-switching wide-area networks. We defined the problem and the type of network to be studied, adopted the real-time channel abstraction, focused our attention on two types of performance guarantees such a service could provide, specified the meaning and extent of these guarantees, and selected a simple characterization of a channel's input packet stream.

A single-round-trip procedure for establishing channels has been devised. The procedure entails several tests and tentative reservations of resources to be performed in each node along the channel's path. A channel that passes these tests can be guaranteed the type and value of delay bound requested by the client; these guarantees become effective as soon as the channel is established and remain in effect until it is disconnected. There is no need to verify that this is true while real-time services are being provided, nor to make on-the-fly adjustments to policies in the nodes or the hosts. Our simulation experiments have failed to disprove the correctness of the scheme even in worst-case situations. Note that performance can be guaranteed because of the scheduling and distributed rate control policies adopted as well as because of the worst-case bounding arguments on which our establishment scheme is based.

Our approach tries to meet the correctness objective by introducing a number of worst-case assumptions. One effect of these worst-case arguments is the incomplete exploitation of the network's resources by the real-time service. However, the maximum node loads in our simulations reached high values in most nodes (see Figure 1). During second phase simulations (i.e., under maximum offered real-time load), the node utilizations averaged over time were for several nodes in the neighborhood of 50%. The utilization of buffer space, however, is not too good for statistical channels, while it is acceptable for deterministic channels. An important point in favor of our approach is that, unlike what happens in the case of circuit switching, network bandwidth, processing power, and buffer space not used by the real-time service are always available to other communication services or local processing tasks.

The CPU overheads of scheduling and distributed rate control seem to be quite reasonable, in the range of a few tens of microseconds. While these are a bit high for an ATM network, with its small cell sizes and high speed transmission links, they appear to be acceptable for packet switching networks, especially with large packet sizes. The execution of the scheduling and rate control algorithms will of course be speeded up in the future by the use of inexpensive high-speed machines as nodes, and even more (actually, much more) by the implementation of these algorithms in hardware, which is made easily feasible by their simplicity. The establishment computations are simple to implement and do not take more than a few milliseconds at each node.

Many problems still remain to be explored in the field of wide-area real-time communication. Some of the following are being investigated, while others will be in the near future:

- the best way to guarantee a given bound on delay variance or delay jitter;
- the possibility of devising correct algorithms for the establishment of channels whose traffic is described by parameters different from x_{min} and x_{ave}, or is dependent on the traffic of other channels;
- the introduction of security, fault tolerance, accounting, and charging capabilities into the design of a real-time service;

7. Acknowledgement

The authors are grateful to David Anderson for proposing the parametrized message channel abstraction which is the basis of the real-time channels discussed in this paper. They are indebted to the members of the DASH and Tenet groups at Berkeley for the many useful comments and suggestions.

8. References

[CoYa88] D. E. Comer and R. Yavatkar, "Flows: Performance Guarantees in Best Effort Delivery Systems", *Proc. 8th Int. Conf. Distrib. Comp. Sys.*, San Jose, CA, pp. 376-383, June 1988.

[DaVe89] S. Damaskos and D. C. Verma, "Fast Establishment of Real-Time Channels", Tech. Rept. TR-89-056, International Computer Science Institute, Berkeley, October 1989.

[Ferr89] D. Ferrari, "Real-Time Communication in Packet Switching Wide-Area Networks", Tech. Rept. TR-89-022, International Computer Science Institute, Berkeley, May 1989.

[Ferr90] D. Ferrari, "Client Requirements for Real-Time Communication", Tech. Rept. TR-90-007, International Computer Science Institute, Berkeley, March 1990.

[FeVe90] D. Ferrari and D. C. Verma, "A Scheme for Real-Time Channel Establishment in Wide-Area Networks", *IEEE J. Selected Areas Communic.*, vol. SAC-8, n. 3, pp. 368-379, April 1990.

[FeVe90b] D. Ferrari and D. C. Verma, "Buffer Allocation for Real-Time Channels in a Packet-Switching Network", Tech. Rept. TR-90-022, International Computer Science Institute, Berkeley, June 1990.

[Harr80] E. Harrington, "Voice/data integration using circuit-switched networks", *IEEE Trans. Comm.*, vol. COM-28, n. 6, pp. 781-793, June 1980.

[Lein89] B. Leiner, "Critical Issues in High Bandwidth Networking", Internet RFC-1077, Nov. 1988.

[LiLa73] C. L. Liu and J. W. Layland, "Scheduling algorithms for multiprogramming in a hard real time environment", *J. ACM*, vol. 20, n. 1, pp. 46-61, January 1973.

[OhON88] H. Ohnishi, T. Okada and K. Noguchi, "Flow control schemes and delay/loss trade-offs in ATM networks", *IEEE J. Selected Areas Communic.*, vol. SAC-6, n. 6, pp. 1609-1616,December 1988.

[Schw89] H. Schwetman, "CSIM Reference Manual (Revision 12)", MCC Tech. Rept. No. ACA-ST-252-87, January 1989.

[SeJo87] K. C. Sevcik and M. J. Johnson, "Cycle time properties

of the FDDI token ring protocol", *IEEE Trans. Softw. Engin.*, vol. SE-13, pp. 376-385, Mar. 1987.

[Zhan87] L. Zhang, "Designing a new architecture for packet switching communication networks", *IEEE Communic. Magaz.*, vol. 25, n. 9, pp. 5-12, September 1987.