MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

Connection Admission Control for Hard Real-Time Communication in ATM Networks

Qin Zheng, Tetsuya Yokotani, Tatsuki Ichihashi, Yasunoni Nemoto

TR96-21 December 1996

Abstract

Connection Admission Control (CAC) is needed in ATM networks to provide Quality of Service (QoS) guarantees to real-time connections. This paper presents a CAC scheme based on a bit-stream traffic model, which is capable of modeling traffic generation patterns of CBR/VBR connections and traffic distortions within a network, and worst-case queueing analysis to obtain cell queueing delay bounds. The proposed CAC scheme can be used to establish hard real-time connections in ATM networks with conventional static priority FIFO queueing switches. The effectiveness of the scheme is illustrated by applying it to RTnet, an ATM-based real-time plant control network currently being developed by the Mitsubishi Electric Corporation. The CAC scheme presented in the paper can also be extended to set up soft real-time connections in ATM networks.

MERL – A MITSUBISHI ELECTRIC RESEARCH LABORATORY
http://www.merl.com

Connection Admission Control for Hard Real-Time Communication in ATM Networks

Qin Zheng
MERL — A Mitsubishi Electric Research Laboratory
Tetsuya Yokotani, Tatsuki Ichihashi
Mitsubishi Electric Information Technology R&D Center
Yasunoni Nemoto
Power and Industrial Systems Center
Mitsubishi Electric Corporation
TR-96-21 September 1996

Abstract

Connection Admission Control (CAC) is needed in ATM networks to provide Quality of Service (QoS) guarantees to real-time connections. This paper presents a CAC scheme based on a bit-stream traffic model, which is capable of modeling traffic generation patterns of CBR/VBR connections and traffic distortions within a network, and worst-case queueing analysis to obtain cell queueing delay bounds. The proposed CAC scheme can be used to establish hard real-time connections in ATM networks with conventional static priority FIFO queueing switches. The effectiveness of the scheme is illustrated by applying it to RTnet, an ATM-based real-time plant control network currently being developed by the Mitsubishi Electric Corporation. The CAC scheme presented in the paper can also be extended to set up soft real-time connections in ATM networks.

1 Introduction

Asynchronous Transfer Mode (ATM) [1] is a promising technology to provide integrated services for high-speed digital communication networks. Its ability to support high bandwidth, high reliability, and guaranteed quality of service communication makes it an ideal network for supporting many distributed real-time applications. However, to make ATM capable of supporting hard real-time communications such as that required by plant control systems, more sophisticated Connection Admission Control (CAC) schemes than conventional ones are needed.

For example, one way to support real-time communication in ATM networks is to use the Constant Bit Rate (CBR) service. The CBR service is provided via rate control at sources which limits the maximum rate that cells of a connection can be injected into a network, and Connection Admission Control (CAC) which limits the number of real-time connections over each transmission link. One straightforward CAC scheme for CBR connections is the peak bandwidth allocation which limits the aggregated bandwidth of all CBR connections over a transmission link not to exceed the link bandwidth. However, due to traffic distortions within a network, peak bandwidth allocation can not guarantee hard cell transmission delay bounds for CBR connections. Specifically, due to cell jitters introduced at upstream nodes, cells of a CBR connection may arrive at a switching node at a higher rate than they are controlled at their source. This fluctuation in cell rate may cause the aggregated cell arrival rate at a switch to exceed the outgoing link bandwidth, resulting in unpredictable queueing delays for CBR cells. Thus to provide hard real-time guarantees for CBR connections, more sophisticated CAC schemes than the peak bandwidth allocation are needed to ensure that the worst-case delays do not exceed the delay bounds requested by the connections.

In addition, CBR is not always the most suitable service for real-time communications. Real-time traffic can also be bursty. It causes over reservation of network resources if one assigns a CBR connection with a bandwidth equal to the peak rate that traffic may be generated. The Variable Bit Rate (VBR) service as defined by the ATM Forum [2] is more suitable for bursty real-time traffic. The traffic model for a VBR connection is described by a peak cell rate (PCR), a sustainable cell rate (SCR), and a maximum bursty size (MBS). A VBR connection is allowed to inject up to MBS cells into a network at a rate of PCR under a constraint that the average cell rate does not exceed SCR. In other words, a connection is allowed to generate traffic at a higher rate for a certain period of time if it did not use up the average bandwidth allocated to it. The VBR service can support bursty real-time communication better than the CBR service, but it is also more difficult to perform connection admission control for real-time VBR connections.

The problem of supporting hard real-time communication in packet or cell switched networks has been studied extensively in recent years [3, 4, 5, 6, 7, 8]. Most results, however, require transmission scheduling and/or shaping mechanisms which have not been implemented in most of the existing ATM switches. The admission control algorithms for the establishment of hard real-time connections over an ATM network with conventional First In First Out (FIFO) queueing and scheduling switches was first studied by Amitava Raha et. al. [9]. In [9], a maximum rate function is used to describe traffic generation patterns at sources and traffic distortions within a network, and the worst-case queueing analysis was performed to obtain the queueing delay bounds for connections.

In this paper, we propose a CAC scheme which uses a similar framework as that of [9] but with following important differences:

More efficient traffic model : a bit-stream traffic model is used which has the advantage of being easier to describe traffic generation pattern of a CBR/VBR connection and traffic distortions within a network.

More accurate modeling of traffic distortions : exact worst-case traffic distortions (rather than an upper bound) within a network can be obtained by using the bit-stream traffic model. Also, the filtering effect of a transmission link to the aggregated traffic of multiple connections can be modeled to obtain tighter queueing delay bounds.

Easier delay bound calculation : explicit algorithms are given for the calculation of worst-case queueing delays which, unlike that of [9], do not require a maximum operation on a complex (and sometimes not easy to obtain) function over a continuous interval. Also, the CAC algorithms proposed in this paper avoid iteration procedures in the delay bound calculation by having each switch provide fixed delay bounds to connections regardless of the current traffic load.

Support for priority scheduling : the CAC scheme presented in this paper allows a switch to assign multiple priority levels for real-time connections so that heterogeneous real-time connections can be accommodated more flexibly.

The CAC scheme proposed in this paper has been applied to RTnet, an ATM-based real-time plant control network under development by the Mitsubishi Electric Corporation, to justify the design of real-time cyclic transmission support using CBR connections and to investigate the feasibility of supporting VBR connections to accommodate bursty real-time traffic. The proposed CAC scheme will also be implemented in RTnet for the management of real-time connections.

This paper is organized as follows. A bit-stream traffic model is introduced in Section 2 for describing CBR/VBR traffic generations at sources. Section 3 presents bit-stream manipulation algorithms for modeling traffic distortions within a network. Queueing delay analysis and CAC algorithms are delivered in Section 4. Section 5 shows the application of the proposed CAC scheme for the RTnet, and the paper concludes with Section 6.

2 Bit Stream Traffic Model

Traffic of a VBR connection is generated at its source with a constraint of the VBR traffic model (PCR, SCR, MBS), where PCR is the peak cell rate, SCR is the sustainable cell rate, and MBS is the maximum burst size. For the convenience of analysis, we assume in this paper that time is measured in a unit of cell time which is defined as the time needed to transmit one cell at the full link bandwidth, and cell rate is normalized to the link bandwidth. Let t_k denote the time that the kth cell of a VBR connection is generated, then t_k satisfies the following inequality:

$$t_k \geq \begin{cases} t_{k-1} + 1/PCR & if\ C_k \geq 1 \\ t_{k-1} + 1/SCR & if\ C_k < 1 \end{cases} \tag{1}$$

where C_k is updated as

$$C_k = \max\{MBS, C_{k-1} + (t_k - t_{k-1}) * SCR - 1\}$$

with $C_0 = MBS$ and t_0 equals to the generation time of the first cell.

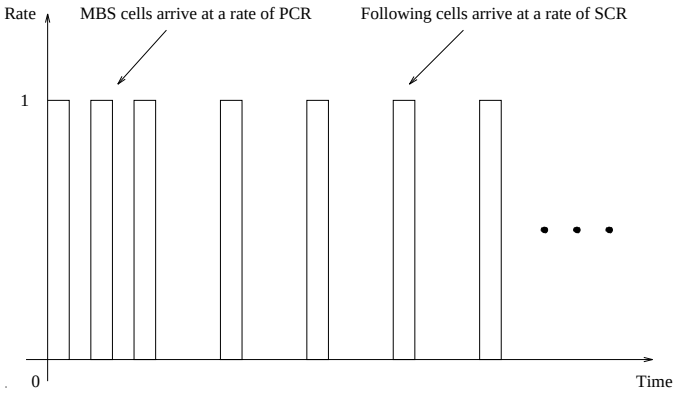

Figure 1: Discrete cell generation model of a VBR connection.

The physical meaning of the above traffic model is that C_k represents the number of tokens that a VBR connection possesses which is decreased by one each time a cell is sent and increased at a rate of SCR up to a maximum value of MBS. A source is allowed to generate cells at a rate of PCR whenever $C_k \geq 1$. Otherwise, the cell generation rate is constrained by the rate at which the token is incremented, i.e., SCR. With this traffic model, a source is allowed to generate a burst of up to MBS cells at a rate of PCR as long as it does not exceed an average transmission rate of SCR. If we define the worst-case traffic generation pattern during a time period $[0, t]$ as the one with which the maximum number of cells is generated during this period, then as shown in Figure 1, the worst-case traffic generation pattern of a VBR connection is that MBS cells are generated at a rate of PCR and after that cells are generated at a rate of SCR continuously.

A CBR connection is described by a peak cell rate PCR which controls the maximum rate that cells can be generated. Since a CBR connection can be viewed as a VBR connection with $SCR = PCR$, we treat CBR as a special case of VBR in this paper.

To perform worst-case queueing analysis, we introduce a bit-stream traffic model to describe the worst-case traffic generation pattern of a VBR connection as shown Figure 2. Specifically, a continuous bit stream is used to approximate the discrete cell generation of a connection in such a way that the bit stream generates the same number of bits as that of the cell stream at cell boundaries. As will be seen later, it is a lot easier to analyze continuous bit streams than discrete cell streams.

A bit stream is described by a bit-stream model

$$S = \{(r(k), t(k)), k = 0, 1, \cdots, m\} \tag{2}$$

which represents the bit rate r as a monotonic decreasing step-wise function of time t as shown in Figure 3. More specifically, S represents a bit stream which has a rate $r(k)$ during a time interval $[t(k), t(k+1))$ for $t = 0, \cdots, m$ with $t(m+1) = \infty$.

From Figure 2, it is easy to see that the conversion from a discrete cell model to a continuous bit-stream model for a VBR connection can be performed with the following algorithm.

Algorithm 2.1 (Conversion to a bit-stream model) .
The worst-case traffic generation of a VBR connection with parameters (PCR, SCR, MBS) *is bounded by the following bit stream*

$$S = \{(1, 0), (PCR, 1), (SCR, 1 + (MBS - 1)/PCR)\} \tag{3}$$

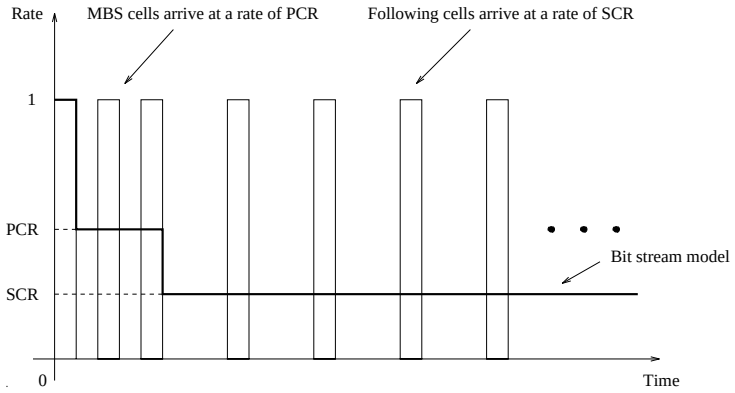

Figure 2: Continuous bit-stream model of a VBR connection.

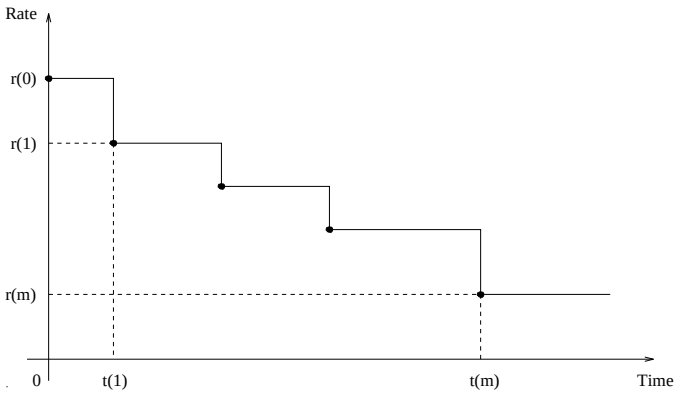

Figure 3: A general bit-stream model.

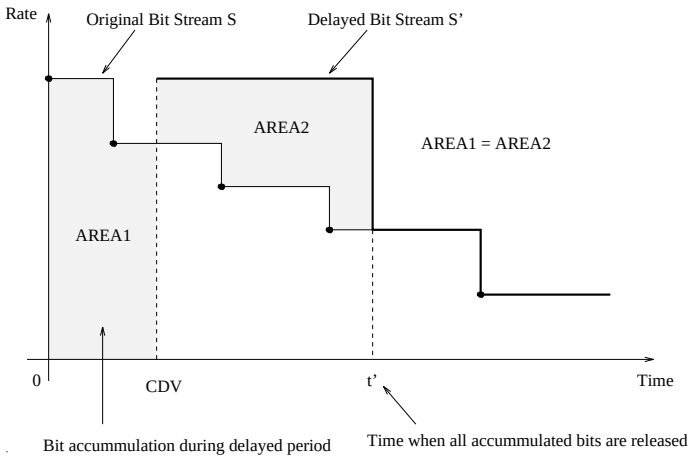

Figure 4: Delay of a bit stream.

3 Bit Stream Manipulation Algorithms

When a bit stream passes through a network, it can be delayed, multiplexed/demultiplexed with other bit streams at queueing points, and filtered by transmission links. This section delivers bit stream manipulation algorithms to model such traffic distortions within a network.

3.1 Delay

Passing a bit stream through a network with a delay variance of CDV can cause clumping of bits. In the worst case, all bits generated during a time period $[0, CDV]$ can be delayed until time CDV and then released at a full link rate, changing the original bit stream S to a delayed bit stream S' as shown in Figure 4.

The conversion from an original stream $S = \{((r(k), t(k)), k = 0, ..., m\}$ to a delayed stream $S' = \{((r'(k), t'(k)), k = 0, ..., m'\}$ can be performed with the following steps:

Step 1: Calculate the bit accumulation during the delayed period $[0, CDV]$ as shown by AREA1 in Figure 4.

Step 2: Calculate time t' when all accumulated bits are release, i.e., when AREA2 equals to AREA1 as shown in Figure 4.

Step 3: Construct a delayed bit stream S' which has a bit rate $r'(t) = 1$ during time period $[0, t' - CDV)$, and bit rate $r'(t) = r(t + CDV)$ for $t \geq t' - CDV$.

The following Pseudo code describes an algorithm implementing the above bit stream conversion steps:

Algorithm 3.1 (Delay of a bit stream) .
Passing a bit stream $S = \{((r(k), t(k)), k = 0, ..., m\}$ through one or more queueing points which have an accumulated maximum delay variation of CDV results in a worst-case delayed bit stream $S' = \{((r'(k), t'(k)), k = 0, ..., m'\}$ which can be calculated as follows:

```
AREA1 = 0
AREA2 = 0
```

```
k = 0                        /* index variable for bit stream S */
k' = 0                       /* index variable for bit stream S' */

/* Step 1: calculation of AREA1 */
while (t(k+1) < CDV)
  AREA1 = AREA1 + r(k)*(t(k+1)-t(k))
  k = k + 1
AREA1 = AREA1 + r(k)*(CDV-t(k))

/* Step 2: calculation of t' */
AREA2 = (1-r(k))*(t(k+1)-CDV)
while (AREA1 > AREA2)
  k = k + 1
  AREA2 = AREA2 + (1-r(k))*(t(k+1)-t(k))
t' = t(k+1) - (AREA2-AREA1)/(1-r(k))

/* Step 3: construction of S' */
t'(0) = 0
r'(0) = 1
t'(1) = t' - CDV
r'(1) = r(k)
k'=1
while (k < m)
  k = k + 1
  k' = k' + 1
  t'(k') = t(k) - CDV
  r'(k') = r(k)
m' = k'
```

Algorithm 3.1 can be used to calculate the worst-case bit stream arrival of a CBR/VBR connection at an incoming link of a switch with a known accumulated maximum delay variance CDV over upstream switches.

3.2 Multiplexing

Suppose two bit streams $S1$ and $S2$ arrive at a switch, then the worst-case multiplexed bit stream rate is the summation of each individual bit stream rate as shown in Figure 5. The following algorithm calculates the multiplexed bit stream $S = S1 + S2$:

Algorithm 3.2 (Bit stream multiplexing) .
Let $S1 = \{(r1(k1), t1(k1)), k1 = 0, \cdots, m1\}$, $S2 = \{(r2(k2), t2(k2)), k2 = 0, \cdots, m2\}$, *then the multiplexed bit stream* $S = S1 + S2 = \{(r(k), t(k)), k = 0, \cdots, m\}$ *can be calculated as follows:*

```
k1 = 1                                  /* index variable for S1 */
k2 = 1                                  /* index variable for S2 */
m = 0                                   /* m for S */
r(0) = r1(0) + r2(0)

/* at each time point t where r1(t) or r2(t) changes, */
```

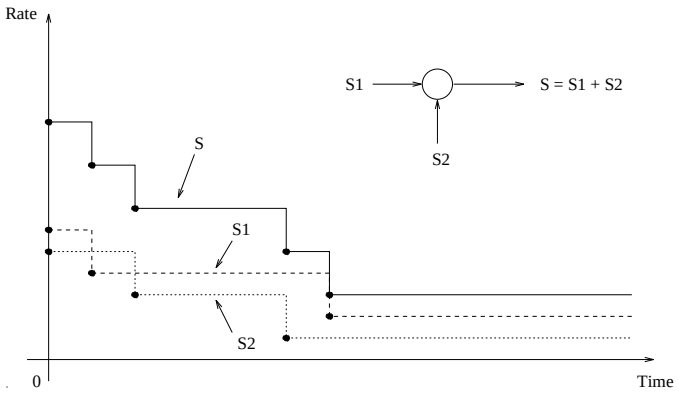

Figure 5: Multiplexing of two bit streams.

```
/* calculate r(t) = r1(t) + r2(t) */
While ((k1 <= m1) & (k2 <= m2))
  m = m + 1
  if (t1(k1) < t2(k2))                    /* r1(t) changes first */
    t(m) = t1(k1)
    r(m) = r1(k1) + r2(k2-1)
    k1 = k1 + 1
  else if (t1(k1) > t2(k2))               /* r2(t) changes first */
    t(m) = t2(k2)
    r(m) = r1(k1-1) + r2(k2)
    k2 = k2 + 1
  else                                    /* both r1(t) and r2(t) change */
    t(m) = t2(k2)
    r(m) = r1(k1) + r2(k2)
    k1 = k1 + 1
    k2 = k2 + 1

if (k1 <= m1)                             /* append tail of S1 */
  for (k1=k1; k1<=m1; k1++)
    m = m + 1
    r(m) = r1(k1) + r2(k2-1)
    t(m) = t1(k1)
else if (k2 <= m2)                        /* append tail of S2 */
  for (k2=k2; k2<=m2; k2++)
    m = m + 1
    r(m) = r1(k1-1) + r2(k2)
    t(m) = t2(k2)
```

3.3 Demultiplexing

Suppose a bit stream $S1$ is aggregated from a bit stream $S2$ and some other bit streams, then the removal of $S2$ from $S1$ results in a new bit stream S whose bit rate $r(t) = r1(t) - r2(t)$ as shown in Figure 6. The calculation of $S = S1 - S2$ can be performed with the following algorithm:

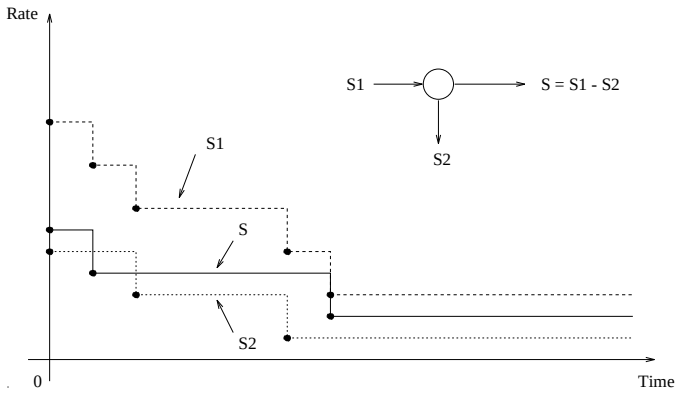

Figure 6: Demultiplexing of two bit streams.

Algorithm 3.3 (Bit stream demultiplexing) .
Let $S1 = \{(r1(k1), t1(k1)), k1 = 0, \cdots, m1\}$, $S2 = \{(r2(k2), t2(k2)), k2 = 0, \cdots, m2\}$, *then the demultiplexed bit stream* $S = S1 - S2 = \{(r(k), t(k)), k = 0, \cdots, m\}$ *can be calculated as follows:*

```
k1 = 1                                      /* index variable for S1 */
k2 = 1                                      /* index variable for S2 */
m = 0                                       /* m for S */
r(0) = r1(0) - r2(0)

/* at each time point t where r1(t) or r2(t) changes, calculate */
/* r(t) = r1(t) - r2(t) */
While ((k1 <= m1) & (k2 <= m2))
  m = m + 1
  if (t1(k1) < t2(k2))                      /* r1(t) changes first */
    t(m) = t1(k1)
    r(m) = r1(k1) - r2(k2-1)
    k1 = k1 + 1
  else if (t1(k1) > t2(k2))                 /* r2(t) changes first */
    t(m) = t2(k2)
    r(m) = r1(k1-1) - r2(k2)
    k2 = k2 + 1
  else                                      /* both r1(t) and r2(t) change */
    t(m) = t2(k2)
    r(m) = r1(k1) - r2(k2)
    k1 = k1 + 1
    k2 = k2 + 1

if (k1 <= m1)                               /* append tail of S1 */
  for (k1=k1; k1<=m1; k1++)
    m = m + 1
    r(m) = r1(k1) - r2(k2-1)
    t(m) = t1(k1)
```

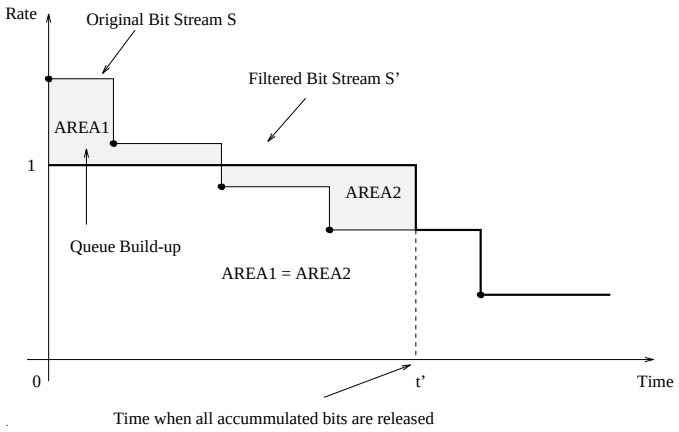

Figure 7: Bit stream filtering.

3.4 Filtering

Multiplexing of several bit streams at a queueing point may result in an aggregated bit rate exceeding the link bandwidth. The outgoing bit stream is thus filtered by the outgoing transmission link as shown in Figure 7. The conversion of an incoming bit stream $S = \{(r(k), t(k)), k = 0, \cdots, m\}$ to a filtered outgoing bit stream $S' = \{(r'(k), t'(k)), k = 0, \cdots, m'\}$ can be performed with the following steps:

Step 1: Calculate the maximum queue buildup during the time period when the incoming bit rate exceeds the outgoing link rate, i.e., when $r(t) > 1$, as shown by AREA1 of Figure 7.

Step 2: Calculate time t' when all accumulated bits are released, i.e., when AREA2 equals to AREA1 as shown in Figure 7.

Step 3: Construct a filtered bit stream S' which has a bit rate $r'(t) = 1$ for $t \in [0, t')$, and bit rate $r'(t) = r(t)$ for $t \geq t'$.

The following algorithm implements the above steps:

Algorithm 3.4 (Bit stream filtering) .
Passing a bit stream $S = \{(r(k), t(k)), k = 0, 1, \cdots, m\}$ through a transmission link with bandwidth of 1 cell per cell time results in a filtered bit stream $S' = filter(S) = \{(r'(k'), t'(k')), k' = 0, 1, ..., m'\}$ which can be calculated as follows:

```
AREA1 = 0
AREA2 = 0
k = 0                                   /* index variable for S */
k' = 0                                  /* index variable for S' */

if (r(k) <= 1) /* bit stream is not changed */
  S' = S
else
  r'(k') = 1
  k'++
```

```
/* step 1: calculation of maximum queue build-up */
while (r(k) > 1)
  AREA1 = AREA1 + (r(k)-1)*(t(k+1)-t(k))
  k = k + 1

/* step 2: calculation of t'(1), the time when the queue becomes empty */
while (AREA1 > AREA2)
  AREA2 = AREA2 + (1-r(k))*(t(k+1)-t(k))
  k = k + 1
t'(k') = t(k) + queue/(1 - r(k-1))
r'(k') = r(k-1)

/* step 3: fill up the rest of S' */
while (k <= m)
  k' = k' + 1
  r'(k') = r(k)
  t'(k') = t(k)
  k = k + 1
```

Traffic filtering by transmission links smooths the incoming bit streams at a queueing point, and thus can greatly reduce the cell queueing delay bounds.

4 Worst-case Queueing Analysis and CAC Algorithm

In this section, we show how the bit-stream model and manipulation algorithms presented in the last section can be used to construct Connection Admission Control (CAC) algorithms for the establishment of CBR/VBR connections in an ATM network.

4.1 Assumptions

Connection admission control is used to determine whether or not a network has enough resources to accommodate new connections. A CAC algorithm depends on (1) the types of QoS parameters requested by a connection; (2) the types of queueing and scheduling mechanisms used at a switching node; and (3) connection setup sequence, i.e., how a CAC algorithm is used within the whole connection setup procedure.

QoS parameters .
To support hard real-time communication in plant control networks, we use the end-to-end queueing delay bound D as the QoS parameter for a CBR/VBR connection. In other words, successful establishment of a connection with parameters (PCR, SCR, MBS, D) means that a network guarantees that cells of this connection will not experience queueing delays larger than D within the network as long as the connection does not inject more cells into the network than that constrained by a peak cell rate PCR, a sustainable cell rate SCR, and a maximum bursty size MBS.

Queueing and scheduling mechanism .
The way in which a switch uses to queue and schedule transmission of incoming cells directly affects a switch's ability of providing QoS guarantees to connections. Sophisticated queueing and scheduling schemes like per-VC queueing and deadline scheduling allow a

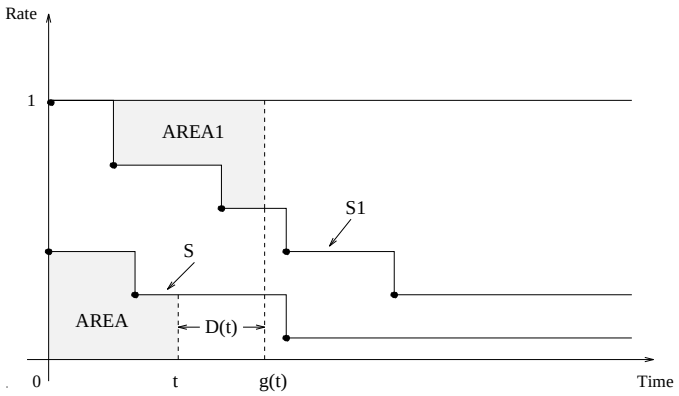

Figure 8: Queueing delay calculation.

switch to provide different QoS guarantees for each individual connections. In this report, however, we assume a basic static priority First-In-First-Out (FIFO) queueing and scheduling scheme to ensure the applicability of our CAC algorithms to existing systems like RTnet. Specifically, with a static priority FIFO scheduling mechanism, incoming cells of a connection are stored in one of the priority FIFO queues. Cells stored in a higher priority queue are always sent before those in a lower priority queue. Within the same queue, cells are sent in the same order as they are deposited into the queue. With this queueing mechanism, connections of the same priority have the same queueing delay bound at the queueing point.

Connection setup sequence .

We assume a distributed connection setup procedure. Specifically, each switching node in a network provides a maximum queueing delay bound D_{max} to CBR/VBR connections whereby D_{max} equals to the size of the FIFO queue for CBR/VBR connection at the switching node. A source end system requests a CBR/VBR connection by sending a SETUP message containing its traffic and QoS parameters, i.e., (PCR, SCR, MBS, D), along a preselected route to the destination. Each switch on the route processes the SETUP message by checking whether or not there are enough resources to accommodate this new connection by executing a CAC algorithm. If the connection passes the CAC check, the switch forwards the SETUP message to a downstream node on the route. Otherwise, the switch sends back a REJECT message notifying upstream switches and the source end system that the requested connection can not be established. If the SETUP message successfully reaches a destination end system, a CONNECTED message is sent back to the source end system notifying the completion of connection setup. The source end system can then start sending data cells.

4.2 Queueing delay bound calculation

With assumptions given above, we first derive an algorithm to calculate the queueing delay bound for a connection of transmission priority p. Let $S = \{(r(k), t(k)), k = 0, \cdots, m\}$ be the aggregated arriving bit stream of priority p, and let $S1 = \{(r1(k1), t1(k1)), k1 = 0, \cdots, m1\}$ be the filtered aggregated arriving bit stream of priority levels higher than p at a queueing point. Then as shown in Figure 8, the queueing delay bound for S can calculated with the following steps:

Step 1: For each time point $t \geq 0$, find a corresponding time point $g(t)$ such that AREA1 equals to AREA. The physical meaning of $g(t)$ is that in the worst case, a bit of S arrived at time t would leave the queueing point at time $g(t)$.

Step 2: Let $D(t) = g(t) - t$, then $D(t)$ is the worst-case queueing delay for a bit of S arrived at the queueing point at time t. Notice that $D(t)$ is a monotonic increasing function in an interval where $r(t) > 1 - r1(g(t))$ and a monotonic decreasing function in an interval where $r(t) < 1 - r1(g(t))$.

Step 3: Find time t' such that $r(t') = 1 - r1(g(t'))$. Since $r(t)$ is a monotonic decreasing function and $1 - r1(g(t))$ is a monotonic increasing function, the maximum of $D(t)$ is reached at a time t'. In other words, the queueing delay bound for S equals $D(t')$.

Since $r(t)$ and $r1(t)$ change only at a finite number of time points, the queueing delay bound can be obtained with a finite number of operations. The following algorithm implements the above steps:

Algorithm 4.1 (Calculation of queueing delay bound) .
Let $S = \{(r(k), t(k)), k = 0, \cdots, m\}$ be the aggregated arriving bit stream of priority p and $S1 = \{(r1(k1), t(k1)), k1 = 0, \cdots, m1\}$ be the filtered aggregated arriving bit stream of priority levels higher than p at a queueing point. Then the queueing delay bound $D = delay\_bound(S, S1)$ for S can be calculated as follows:

```
D = 0                                          /* queueing delay bound */
k = 0                                          /* index for S */
k1 = 0                                         /* index for S1 */
AREA = 0
AREA1 = 0

while (r(k) > 1-r1(k1))                       /* no need to check D(t) otherwise */
  AREA = AREA + r(k)*(t(k+1)-t(k))
  /* now find g(t) for t=t(k) */
  while ((AREA > AREA1) & (r(k) > (1-r1(k1))) & (k1 < m1))
    AREA1 = AREA1 + (1-r1(k1))*(t1(k1+1)-t1(k1))
    k1 = k1 + 1
  if (AREA <= AREA1)
    g(t) = t1(k1) - (AREA1-AREA)/(1-r1(k1-1))
    k = k + 1
    D = g(t) - t(k)
  else                                         // either r(k)<=1-r1(k1) or k1=m1
    k=k+1
    g(t) = t(k) - (AREA-AREA1)/r(k-1)
    D = g(t) - t(k)
    break                                      // maximum D found
```

Notice that if p is the highest priority level, $r1(t) = 0$ for all $t \geq 0$. Then the maximum queueing delay can be simply calculated as AREA1 as shown in Figure 7 for the bit stream S.

4.3 Connection Admission Control Algorithm

To perform the worst-case queueing analysis as described above, the following information needs to be kept for each switching node:

- Traffic and QoS parameters for each connection: (PCR, SCR, MBS, CDV), where PCR, SCR, MBS are the peak cell rate, sustainable cell rate, and maximum bursty size as defined in Section 2, and CDV is the maximum accumulated cell delay variance over upstream nodes.
- For each pair of incoming link i and outgoing link j, and each priority level p of outgoing link j, store the following bit streams:
  - $S_{ia}(i,j,p)$: the aggregated bit stream of all connections of priority p coming in from incoming link i and going out from outgoing link j;
  - $S_{ia}(i,j)(p)$: the aggregated bit stream of all connections of priority levels higher than p coming in from incoming link i and going out from outgoing link j;
  - $S_{if}(i,j,p)$: filtered bit stream of $S_{ia}(i,j,p)$;
  - $S_{if}(p)$: filtered bit stream of $S_{ia}(i,j)(p)$;
  - $S_{oa}(j,p)$: aggregated bit stream of $S_{if}(i,j,p)$ over all incoming links i;
  - $S_{oa}(j)(p)$: aggregated bit stream of $S_{if}(i,j)(p)$ over all incoming links i;
  - $S_{of}(j)(p)$: filtered bit stream of $S_{oa}(j)(p)$

With fixed numbers of incoming/outgoing links and priority levels of a switch, the memory needed to store the above data structures is proportional to the number of connections established over the switch.

Assume that a switch supports a queueing delay bound of $D(j,p)$ for outgoing link j and transmission priority p, and let (PCR, SCR, MBS, CDV) be the traffic parameters of a new connection with incoming link i, outgoing link j, and priority level p, then the connection can be established at the switch if and only if the addition of this connection does not affect the switch's delay bound guarantees. This delay bound check can be performed using Algorithm 4.1 with the following steps:

Step 1: Calculate the arrival bit stream S for the new connection using Algorithm 2.1 and Algorithm 3.1 with the connection's traffic parameters (PCR, SCR, MBS, CDV).

Step 2: Calculate a new aggregated arriving bit stream at the incoming link i: $S'_{ia}(i,j,p) = S_{ia}(i,j,p) + S$, and a new filtered aggregated bit stream $S'_{if}(i,j,p) = filter(S'_{ia}(i,j,p))$ using Algorithm 3.2, and Algorithm 3.4, respectively.

Step 3: Calculate a new aggregated bit stream at the outgoing link j: $S'_{oa}(j,p)$, which can be obtained by first subtracting $S_{if}(i,j,p)$ from $S_{oa}(j,p)$ and then adding $S'_{if}(i,j,p)$ using Algorithm 3.2 and Algorithm 3.3, respectively.

Step 4: Calculate the queueing delay bound $D'(j,p) = delay\_bound(S'_{oa}(j,p), S_{of}(j,p))$ for priority p at outgoing link j using Algorithm 3.1. If $D'(j,p)$ is larger than the switch's delay bound $D(j,p)$, stop, the connection can not be established at the switch. Otherwise, go to Step 5 to check if the new connection affects connections of lower priority levels (connections of higher priority levels will not be affected by the new connection).

Step 5: For each priority level p_1 which is lower than p and assigned to real-time connections, calculate the queueing delay bound $D'(j, p_1)$ at outgoing link j using steps described below:

- Calculate a new a new aggregated bit stream $S'_{oa}(j)(p_1)$ by first subtracting $S_{if}(i, j, p)$ from $S_{oa}(j)(p_1)$ and then adding $S'_{if}(i, j, p)$ using Algorithm 3.2 and Algorithm 3.3, respectively.
- Calculate a filtered bit stream of $S_{of}(j)(p_1) = filter(S_{oa}(j)(p_1))$ using Algorithm 3.4.
- Calculate a new queueing delay bound for priority p_1 using Algorithm 3.1: $D'(j, p_1) = delay\_bound(S_{oa}(j, p_1), S_{of}(j)(p_1))$.

Step 6: If $D'(j, p_1) > D_(j, p)$, the connection can not be established over the switch. Otherwise, the connection passes the CAC check at the switch.

Some discussions about the above CAC scheme:

1. In the above CAC scheme, an accumulated cell delay variance CDV is needed to calculate the worst-case arrival bit stream of a connection. For hard real-time connections, this CDV can be calculated as the summation of maximum queueing delays over upstream switches, which represents the worst-case variation of queueing delays that cells of connection may experience. This worst case, however, is very unlikely to happen in practice since the probability of a cell's having maximum queueing delays over all switches on its route is very small. Thus for soft real-time connections, a less conservative CDV accumulation scheme such as square-root summation of queueing delay bounds over upstream switches can be used to accommodate more real-time connections in a network. Comparison of hard and soft CAC schemes will be given in Section 5.

2. The proposed CAC scheme supports multiple priority levels for real-time connections. The advantage of using multiple priority levels is that connections with diverse delay bound requirements can be supported more efficiently (i.e., connections requesting large delay bounds can be assigned low priority levels). However, the computation and memory required to perform the CAC check also increase proportionally with the number of priority levels used to support real-time connections. So for networks where fast establishment of switched real-time VCs is needed, the number of priority levels assigned for supporting real-time traffic should not be too large.

3. The proposed CAC scheme can be implemented either distributedly at switches or centrally at a connection admission control server. For permanent real-time connections, the CAC check can also be performed off-line for which case the memory and computation requirements would not be a big issue.

5 Connection Admission Control for RTnet

The Real-Time Industrial Control Network (RTnet), currently being developed at the Power and Industrial Systems Center of Mitsubishi Electric Corporation, is an ATM-based high speed local area network designed for next generation plant control systems. The CAC scheme presented in this paper has been used in the design phase of RTnet to verify the network's ability of supporting real-time communication. Specifically, the proposed CAC scheme has contributed to (1) validate the way in which real-time cyclic transmission is supported, (2) investigate

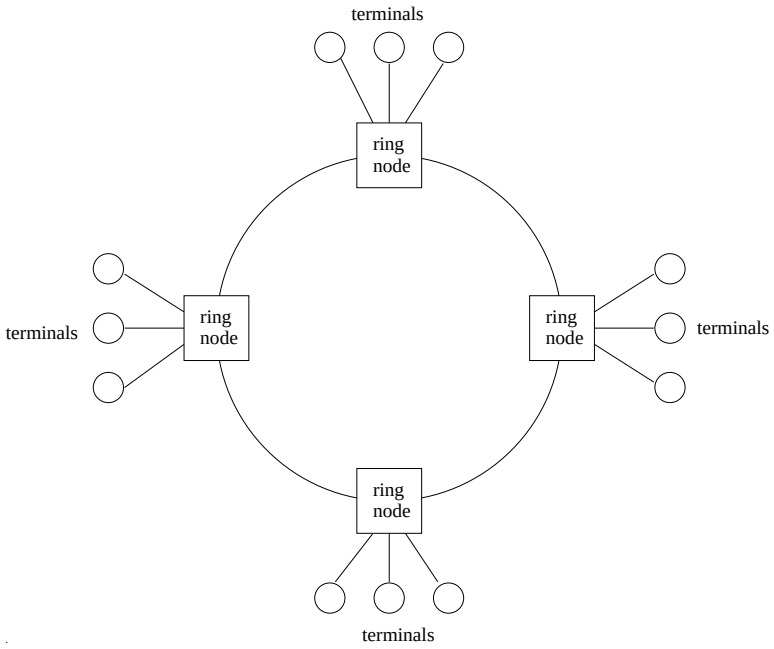

Figure 9: RTnet topology.

	period (ms)	delay (ms)	memory size (KB)	bandwidth (Mbps)
high speed	1	1	4	32
medium speed	30	30	64	17.5
low speed	150	150	128	6.8

Table 1: Types of cyclic transmission.

the feasibility of supporting real-time VBR service, and (3) determine buffer requirement at switched for real-time traffic. The CAC scheme will also be implemented in the next version of RTnet for the management of switched real-time connections. In this section, we present some evaluation results showing how the proposed CAC scheme can be applied to a real world system such as RTnet.

As shown in Figure 9, a typical configuration of RTnet employs a star-ring topology where ring nodes are connected together via dual 155 Mbps links and multiple end terminals are attached to each ring node. One advantage of this configuration is that the network can tolerate any single link/node failure by a using hardware ring wrap-around technology similar to that used in FDDI networks.

An important real-time application supported by the RTnet is cyclic transmission which implements a kind of real-time shared memory among terminals in a network. Specifically, each terminal uses the cyclic transmission facility to periodically broadcast its portion of shared memory to the network and receives updates of other portions of the shared memory from other terminals. Table 1 shows three types of cyclic transmissions that are supported by the RTnet which lists the memory update period, maximum allowable update delay, maximum size of the shared memory, and the calculated maximum bandwidth required for each type of cyclic transmission.

To investigate whether or not an RTnet has enough capacity to support cyclic transmissions with the CBR service, we applied our CAC scheme to a typical RTnet with 16 ring nodes and

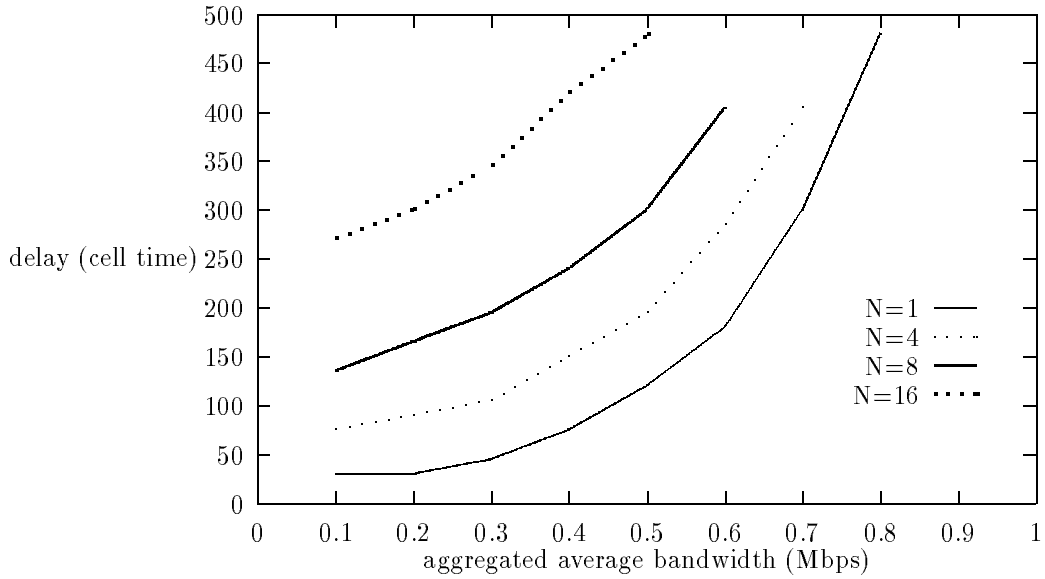

Figure 10: End-to-end queueing delay bounds.

each ring node has N terminals attached to it. The maximum number of terminals that can be connected to a ring node 16. Each node has a highest priority FIFO queue of 32 cells to accommodate cyclic traffic. At a 155 Mbps transmission speed, one cell time is about 2.7 microseconds. Thus a 32-cell FIFO queue represents a maximum of $32 \times 2.7 = 87$ microseconds of queueing delay at each node. In other words, each ring node contributes a maximum of 87 microseconds of CDV to connections established through it.

A symmetric traffic generation pattern is first investigated where each terminal generates the same amount of traffic (i.e., the cyclic memory is equally divided among terminals) with an total normalized traffic load B. The hard CAC scheme presented in Section 4.3 is used to setup a broadcast CBR connection with a $PCR = B/16N$ for each terminal. The obtained maximum end-to-end queueing delay bounds as a function of traffic load B and the number of terminals per node N are plotted in Figure 10 from which we can conclude the following:

- For $N = 1$, up to 75% of cyclic traffic (115 Mbps) can be supported with end-to-end queueing delays smaller than 370 cell times (1 ms). This is much more than the targeted maximum amount of cyclic traffic to be supported in RTnet. As the number of terminals attached to each ring node N increases, the traffic generated by each ring node becomes more bursty, thus resulting in larger queueing delays and less cyclic traffic that can be supported by the network. With a maximum configuration of $N = 16$ terminals per ring node, about 35% of cyclic traffic (55 Mbps) can be supported with an end-to-end queueing delay bound of 370 cell times. This is enough to accommodate the high speed cyclic traffic. For a network with smaller numbers of ring nodes and/or terminals, all three types of cyclic traffics can be supported with a single transmission priority level.

- Notice that the worst-case aggregated traffic from N CBR connections with a peak cell rate R is the same as that of a VBR connection with $PCR = N$, $SCR = N * R$, and $MBS = N$. Thus from Figure 10 we can also learn the RTnet's ability of supporting

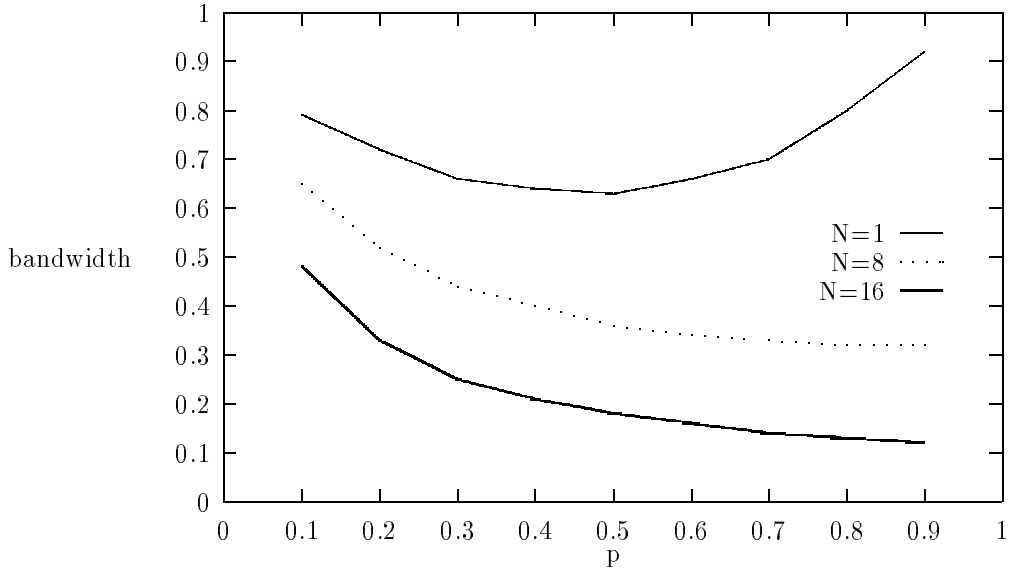

Figure 11: Asymmetric cyclic traffic support.

VBR connections. For example, from the "$N = 16$" curve, it can be concluded that up to 35% of real-time VBR traffic can be supported with a queueing delay bound of 370 cell times if the summation of $MBS's$ of VBR connections established at terminals attached to a ring node does not exceed 16. This shows that it is quite feasible to support the real-time VBR service in RTnet using the proposed CAC scheme.

To study RTnet's ability of supporting asymmetric traffic, we let one terminal generate p% of the total traffic and divide the rest traffic equally among other terminals. The total amounts of cyclic traffic that can be supported by the network as functions of p and the number of terminals per node N are plotted in Figure 11. From the figure we see that the total amount of real-time traffic that can be supported by a network varies with traffic generation patterns. In general, less traffic can be support when the traffic generation pattern becomes more asymmetric (larger p) and more bursty (larger N). In the current version of RTnet, all real-time connections are permanent connections. Thus the proposed CAC algorithm are used to set up real-time connections off-line. The outcomes of the CAC check also help to set network parameters such as ring node buffer sizes and number of priority levels needed to support a given set of real-time connections. For the next version of RTnet, in which switched real-time connections will be supported, the proposed CAC scheme is to be implemented in a central connection management server to set up and tear down real-time connections on-line.

We also evaluated the benefits of using multiple priority levels for supporting real-time traffic. Figure 12 shows the extra amount of real-time traffic that can be supported by an RTnet if two priority levels are used for cyclic traffic.

Finally, Figure 13 shows the extra amount of real-time traffic that can be supported if we use a soft CAC scheme as discussed in Section 4.3.

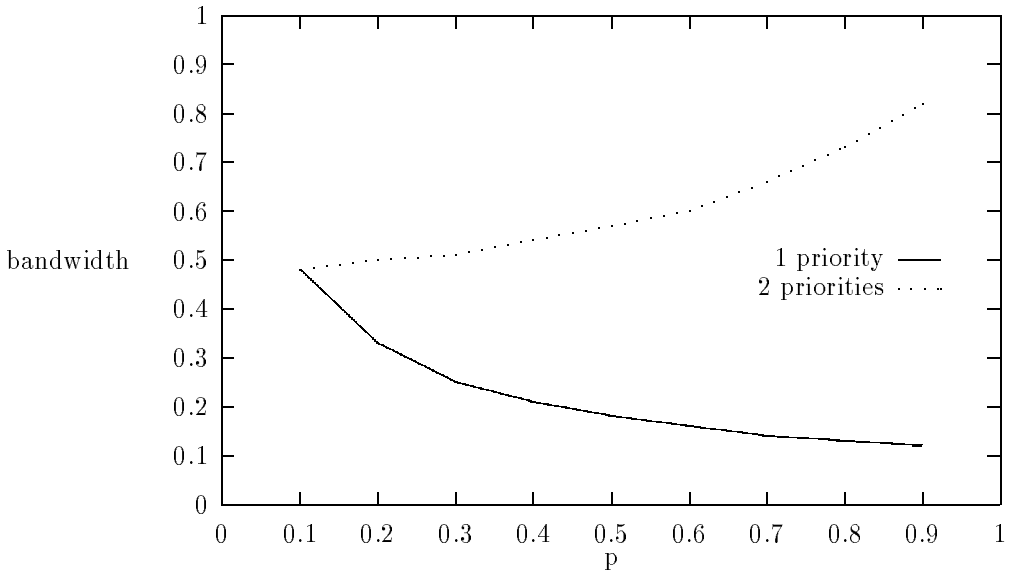

Figure 12: Asymmetric cyclic traffic support with two priority levels.

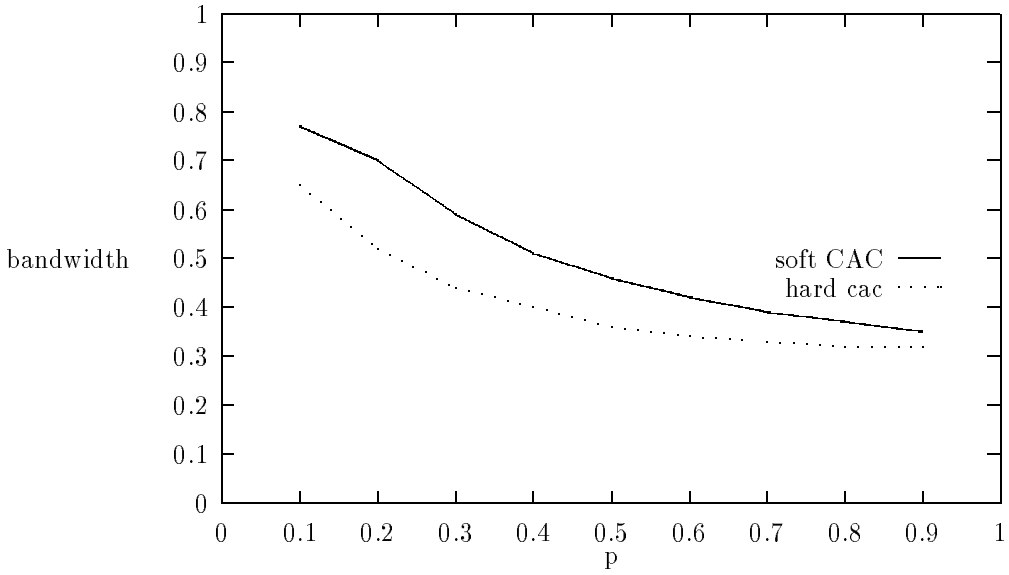

Figure 13: Comparison of soft and hard CAC schemes.

6 Conclusions

We presented in this paper a bit-stream based connection admission control scheme for the establishment of hard real-time connections in ATM networks with conventional static priority FIFO switches. The proposed scheme has been used to verify the design of RTnet, an ATM-based real-time network for industrial control systems, and it is going to be implemented in the next version of RTnet for the management of real-time connections.

References

[1] M. Prycker, *Asynchronous transfer mode: solution for broadband ISDN*, Ellis Horwood Limited, Chichester, West Sussex, PO191EB, England, 1991.

[2] The ATM Forum, "Traffic Management Specification,", April 1996. Version 4.0.

[3] D. Ferrari and D. C. Verma, "A scheme for real-time channel establishment in wide-area networks,", vol. SAC-8, no. 3, pp. 368–379, April 1990.

[4] Q. Zheng and K. G. Shin, "On the ability of establishing real–time channels in point–to–point packet–switched networks," *IEEE Transactions on Communication*, pp. 1096–1105, March 1994.

[5] L. Zhang, "Virtual Clock: A new traffic control algorithm for packet-switched networks," *ACM Trans. Computer Systems*, vol. 9, no. 2, pp. 101–124, May 1991.

[6] A. K. J. Parekh, *A generalized processor sharing approach to flow control in integrated services networks*, PhD thesis, Massachusetts Institute of Technology, February 1992.

[7] Q. Zheng, "An enhanced timed-round-robin traffic control scheme for atm networks," *Proceedings of the 21st Local Computer Network Conference*, November 1996.

[8] H. Zhang and S. Keshav, "Comparison of rate-based service discipline," *Proceedings of ACM SIGCOMM, Zurich, Switzerland*, pp. 113–121, September 1991.

[9] A. Raha, S. Kamat, and W. Zhao, "Admission control for hard real-time connections in ATM LANs," *Proceedings of IEEE INFOCOM'96*, pp. 180–188, April 1996.