

Bandwidth Scheduling for Wide-Area ATM Networks Using Virtual Finishing Times

Anthony Hung, *Student Member, IEEE*, and George Kesidis, *Member, IEEE*

Sept 1994, Revised Sept 1995.

To appear in *IEEE/ACM Trans. Networking*, Feb. 1996

Abstract— This paper is concerned with the design of a class of bandwidth scheduling policies that are suitable for public, wide-area ATM networks. We specify design goals for such strategies including ease of implementation and the ability to guarantee minimum bandwidths to individual buffers. Packetized generalized processor sharing is briefly discussed and a minimum-bandwidth result for self-clocked fair queuing is given. We revisit an approach originally proposed by L. Zhang and prove that it is appropriate for ATM. Some novel, related approaches are described and analyzed.

I. INTRODUCTION

Broadband Integrated Services Digital Network (B-ISDN) has been recognized in the telecommunications community as the all purpose network of the future. B-ISDN is a very high speed network which can carry voice, video, data, etc.; i.e., many different types of traffic with very different required qualities of service (QoS) are handled by a single integrated network. This places a demand on switching and buffering methodologies to satisfy the requested QoS and to use the available bandwidth efficiently. Asynchronous Transfer Mode (ATM) specifies a structure for small, fixed-length (53 byte) packets called cells and a mode of transport for B-ISDN [3], [7].

Our model for a public, wide-area ATM network consists of output buffered switches with non-blocking switch fabrics. Each output port (or “node”) of a switch, shown in Figure 1, is organized as parallel first-in-first-out (FIFO) buffers which share the output link. ATM connections can be grouped into three categories: variable bit rate (VBR), constant bit rate (CBR), available bit rate (ABR), and unspecified bit rate (UBR). Bandwidth *guarantees* are required by VBR, CBR and a certain subclass of ABR connections (with minimum cell rate (MCR) > 0). Typically, connections of the same type will share FIFO buffers. In Figure 1, there are N FIFO buffers with bandwidth guarantees and two best-effort FIFO buffers: one for UBR and the other for the other subclass of ABR connections (MCR=0).

This paper is concerned with the design of suitable algorithms that determine how the buffers share the server, i.e., bandwidth scheduling policies. We focus on algorithms that use virtual finishing times because of their ability to achieve the design goals described in §2. In particular, we define a frame of reference, called the “minimum-

bandwidth” property, which can be used to compare the performance of any two bandwidth scheduling policies.

In §3, we give a more detailed description of the queuing model that will be used subsequently; in particular, we consider bandwidth schedulers based on virtual finishing times and we assume that a cut-through mechanism is in place.

§4 is concerned with proving minimum-bandwidth properties for bandwidth schedulers. Packetized generalized processor sharing is briefly discussed and a minimum-bandwidth result for self-clocked fair queuing is proved. We also prove a minimum-bandwidth property for Virtual Clock; this result was independently obtained by Xie and Lam [14] and Figueira and Pasquale [6] as well. Finally, an “idling” Virtual Clock bandwidth scheduler is introduced and analyzed.

In §5, we re-examine our cut-through assumption; in an elementary way, one can find a minimum-bandwidth property for a non-cut-through implementation of a bandwidth scheduler given its minimum-bandwidth property for a cut-through implementation.

We conclude with a summary in §6.

II. BANDWIDTH SCHEDULING DESIGN GOALS

We now describe our design goals for bandwidth scheduling policies in the context of ATM-based B-ISDN.

First note that the presence of connections requiring deterministic QoS implies the need to be able to guarantee a minimum service bandwidth to any buffer.

ATM has foreseeable transport rates which could be in the order of gigabits per second. Computation speeds have not advanced at such a rapid rate, however. Consequently, the policies used must be simple to implement.

In addition, the number and kinds of connections using a processor sharing node may fluctuate greatly. For efficient bandwidth usage, in response to connection set-up and termination, a policy should be able to easily modify its bandwidth guarantees with a fine degree of bandwidth “granularity”.

These first three goals are most important. Of lesser importance is how the “idle bandwidth” is distributed among nonidle buffers. Roughly speaking, when buffers are idle, the bandwidth allotted to them can be temporarily used elsewhere in the node; we call this resource the idle bandwidth of the node. Since deterministic QoS connections experience no congestion in the network backbone, there is no motive to increase their bandwidth allotment. The idle

bandwidth could be distributed among nonidle (statistical QoS) buffers in some “fair” manner. Alternatively, it may be desirable to use the idle bandwidth to alleviate congestion when it arises or to provide the best-effort traffic with a more regular bandwidth allotment.

A. Definition of the Minimum-Bandwidth Property

In this paper we focus on how bandwidth scheduling policies provide a minimum service bandwidth to the individual FIFO buffers of a processor sharing node. To this end, “minimum bandwidth” is now precisely defined. Buffers of infinite size are assumed.

Consider a bank of N synchronized, discrete-time FIFO buffers sharing a single server with a service capacity of one cell per unit time. If the transmission speed is 150 Mbps, the “unit of time” is $u := 53 \times 8 / (150 \times 10^6) = 2.8 \mu\text{s}$. In the following, all time quantities are in terms of these units and all bandwidth quantities are in terms of cells per unit time.

Let c_i^n denote the i^{th} cell arriving to buffer n and let $a_i^n \geq 0$ be the arrival time of c_i^n to the processor sharing node. Now fix $n \in \{1, 2, \dots, N\}$. Let $\{\mathcal{F}_i^n \mid i \in \{1, 2, \dots\}\}$ be the *Virtual Clock virtual finishing times* (VC-VFTs) based on an arrival process $\{a_i^n \mid i \in \{1, 2, \dots\}\}$ and a bandwidth allotment of ϕ_n cells per unit time. The VC-VFTs are determined by the following recursion:

$$\begin{aligned} \mathcal{F}_i^n &= \max\{\mathcal{F}_{i-1}^n, a_i^n\} + \frac{1}{\phi_n} \quad \text{with} \\ \mathcal{F}_0^n &= 0. \end{aligned} \quad (1)$$

Note that $[\mathcal{F}_i^n]$ is the *departure* time of c_i^n from a buffer having *same* arrival process ($\{a_i^n\}$) as the n^{th} FIFO buffer of the processor sharing node but with a *devoted* server having a bandwidth of *exactly* ϕ_n cells per unit time. Indeed, if A_m^n is the total number of cells¹ arriving to the n^{th} FIFO buffer at time $m \in \mathbb{Z}^+$, then the following recursion determines the number of cells, Y_m^n , in the FIFO buffer with devoted server at time m :

$$\begin{aligned} Y_m^n &= Y_{m-1}^n + A_{m-1}^n - \mathcal{D}_m^n \quad \text{with} \\ Y_{-1}^n &= 0 \end{aligned} \quad (2)$$

where the departure indicators $\mathcal{D}_m^n \in \{0, 1\}$ are such that the departure of the i^{th} cell is $[\mathcal{F}_i^n]$.

Consider a bandwidth scheduler acting on the N FIFO buffers. Let d_i^n be the departure time of c_i^n from the processor sharing node using this scheduler. We say that this bandwidth scheduler has a *minimum-bandwidth property with parameter* $\mu \in \mathbb{Z}$ if

$$d_i^n \leq [\mathcal{F}_i^n] + \mu.$$

for *all* cell arrival processes $\{a_i^n \mid n \in \{1, 2, \dots, N\}, i \in \{1, 2, \dots\}\}$, *all* $n \in \{1, 2, \dots, N\}$, and *all* $i \in \{1, 2, 3, \dots\}$. We can think of these “isolated” buffers with devoted servers as a reference system (\mathcal{F}_i^n) that the bandwidth scheduler of

the processor sharing node (d_i^n) must track. The minimum-bandwidth property is related to the minimum backlog clearing rate (ϕ) of [11] and the service curves of [2].

III. THE QUEUEING MODEL BASED ON VIRTUAL FINISHING TIMES

We now describe our queueing model for the processor sharing node in more detail. Consider again a processor sharing node of the previous section. Cells arriving to the same buffer at the same time are arbitrarily ordered before joining the FIFO buffer. Let $F_i^n \in \mathbb{R}$ be the *virtual finishing time* (VFT) of c_i^n . For all $n \in \{1, 2, \dots, N\}$ and $i \geq 1$, the VFTs are determined recursively as follows:

$$\begin{aligned} F_i^n &= \max\{F_{i-1}^n, v(a_i^n)\} + \frac{1}{\phi_n} \quad \text{with} \\ F_0^n &= 0 \end{aligned} \quad (3)$$

where v is *virtual time* function of the node and $\phi_n > 0$ is the bandwidth partitioning parameter for buffer n . That is, each buffer n is to be guaranteed a minimum service bandwidth of ϕ_n cells per unit time with $\phi_1 + \phi_2 + \dots + \phi_N \leq 1$ (this sum may be temporarily < 1 because ABR flow control policies cannot immediately react to bandwidth availability when, for example, a connection terminates). The order in which cells depart is typically determined by the numerical order of their VFTs. If some head-of-buffer cells have the same VFT, then those cells are served in arbitrary order. We assume that a “cut-through” mechanism is in effect so that a cell arriving at time t to a node may depart at time t as well (experiencing no queueing delay), c.f. §5.

The number of cells X_m^n in the n^{th} FIFO buffer of the processor sharing node at time m is given by the following recursion:

$$\begin{aligned} X_m^n &= X_{m-1}^n + A_{m-1}^n - D_m^n \quad \text{with} \\ X_{-1}^n &= 0 \end{aligned} \quad (4)$$

where the departure indicators $D_m^n \in \{0, 1\}$ are such that $\sum_{n=1}^N D_m^n \in \{0, 1\}$ for all $m \in \mathbb{Z}^+$.

IV. MINIMUM-BANDWIDTH PROPERTY RESULTS

In this section, three previously proposed work-conserving² (nonidling) bandwidth scheduling policies (PGPS, SCFQ and Virtual Clock) are described and their minimum-bandwidth properties are found. Also, a novel bandwidth scheduling policy (“idling” Virtual Clock) is introduced and its minimum-bandwidth property is found as well.

A. Packetized Generalized Processor Sharing

Generalized processor sharing (GPS) is a scheduling algorithm based on an idealized fluid model for traffic flow. Under GPS, if buffer i is nonempty at (continuous) time t , the amount of instantaneous bandwidth it receives is

²Best-effort cells are eligible to depart only when all N buffers with bandwidth guarantees are idle. Best-effort ABR cells take priority over best-effort UBR cells.

¹Note that $\sum_{m=0}^{a_i^n-1} A_m^n < i \leq \sum_{m=0}^{a_i^n} A_m^n$.

$\phi_i / \sum_{j \in A(t)} \phi_j$ where $A(t)$ is the index set of nonempty buffers at t . So, roughly speaking, GPS distributes the idle bandwidth at time t in proportion to the ϕ of the nonempty buffers; this is what we call *GPS fairness*.

Packetized GPS (PGPS) [5], [12] is a work-conserving bandwidth scheduling policy using the VFTs of §3 with a virtual time derived from GPS. A cell arriving to a PGPS node is given a VFT so that, given no further cells arrive, its order of departure is the same as that under the GPS policy.

For a cut-through implementation, PGPS has a minimum-bandwidth property with parameter $\mu = -1$ (Theorem 1 of [12]). Also, the effective bandwidth result of [4] can allow the network to account for the added capacity due to the idle bandwidth scheduling (the problem is that the traffic descriptors required may be difficult to compute and police).

B. Self-Clocked Fair Queuing

A virtual time function that is less complex than PGPS was proposed in [8], [13]: $v(t)$ is the VFT of the last non-best-effort cell served *before* time t ; if no cell has been served before time t , $v(t) = 0$ (this differs slightly from the definition in [8] to accommodate cut-through). That is,

$$v(t) = \begin{cases} 0 & \text{if } \{(n, i) \mid d_i^n < t\} = \emptyset \text{ else} \\ F_i^n & \text{where } (n, i) = \arg \sup \{d_i^n \mid d_i^n < t\} \end{cases}$$

Golestani reasoned that this approach, called self-clocked fair queuing (SCFQ), does approximately provide GPS fairness. We now explore how SCFQ guarantees minimum bandwidths.

Theorem 1: SCFQ has a minimum-bandwidth property with parameter $\mu = N - 2$.

Proof:

For an arbitrary cell c_i^n , let k be the largest integer such that $d_{k-1}^n < a_k^n$ and $2 \leq k \leq i$; if no such k exists set $k = 1$. This means that c_k^n is the last cell to arrive at an empty buffer n before or at time a_i^n in the SCFQ node.

We now show that, if $k < i$,

$$F_j^n = F_{j-1}^n + \frac{1}{\phi_n} \quad \forall j \in \{k+1, k+2, \dots, i\}. \quad (5)$$

The definition of k implies that c_j^n arrives to a nonempty buffer (i.e., $d_{j-1}^n \geq a_j^n$). Since v is nondecreasing (Lemma 2 of [8]), $v(a_j^n) \leq F_{j-1}^n$. Therefore, Equation (5) follows directly from Equation (3).

We now show that

$$F_k^n = v(a_k^n) + \frac{1}{\phi_n}. \quad (6)$$

This is clearly true for $k = 1$. For $k > 1$, note that c_{k-1}^n is served before a_k^n . So, since v is nondecreasing, $v(a_k^n) \geq F_{k-1}^n$. Equation (6) then follows directly from Equation (3).

By Equations (5) and (6),

$$F_i^n = v(a_k^n) + \frac{i - k + 1}{\phi_n}. \quad (7)$$

Because c_i^n arrives during a busy period initiated by c_k^n , precisely $i - k$ cells depart SCFQ buffer n over $[a_k^n, a_i^n]$. Therefore,

$$d_i^n \leq a_k^n + i - k + \sum_{m \neq n} \delta_m \quad (8)$$

where δ_m is the maximum number of buffer m cells that can depart the SCFQ node in $[a_k^n, a_i^n]$; all such cells must have VFTs less than or equal to F_i^n and greater than or equal to $v(a_k^n)$.

Since $F_{j+1}^m - F_j^m \geq \phi_m^{-1}$ for all $j \geq 1$ and $m \in \{1, 2, \dots, N\}$,

$$v(a_k^n) + \frac{\delta_m - 1}{\phi_m} \leq F_i^n = v(a_k^n) + \frac{i - k + 1}{\phi_n}.$$

where the equality is Equation (7). The " $v(a_k^n)$ " terms in this inequality cancel out giving $\delta_m \leq 1 + (i - k + 1)\phi_m / \phi_n$. By substituting into Equation (8) we get

$$d_i^n \leq a_k^n + i - k + \sum_{m \neq n} (1 + (i - k + 1)\frac{\phi_m}{\phi_n}) \quad (9)$$

$$\leq a_k^n + i - k + N - 1 + (i - k + 1)\frac{1 - \phi_n}{\phi_n} \quad (10)$$

$$= a_k^n + \frac{i - k + 1}{\phi_n} + N - 2. \quad (11)$$

Clearly, by Equation (1), for all $j \in \{1, 2, \dots, i\}$,

$$\mathcal{F}_i^n \geq a_j^n + \frac{i - j + 1}{\phi_n}. \quad (12)$$

Subtracting this equation with $j = k$ from Equation (11), we see that

$$d_i^n \leq \mathcal{F}_i^n + N - 2 \leq \lceil \mathcal{F}_i^n \rceil + N - 2. \quad (13)$$

To reiterate, Theorem 1 gives a bound on how much the delay of a cell passing through a SCFQ node can be greater than the delay through the corresponding reference FIFO buffer describe by Equations (1) and (2); in particular, the difference can be on the order of N units of time.

We now offer an example where a cell departs the SCFQ node *after* it would have departed the corresponding isolated buffer. Suppose that there are $N \geq 4$ buffers each with $\phi_n = N^{-1}$ for all $n = 1, 2, \dots, N$ and all buffers are initially empty. Consider the following arrival stream: $a_1^n = 0$ and $a_2^n = 1$ for all $n = 1, 2, \dots, N - 1$, and $a_1^N = 1$. Consequently, $F_1^n = N$ and $F_2^n = 2N$ for all $n = 1, \dots, N - 1$, and $F_1^N = 2N$. Therefore, the departure time of c_1^N under SCFQ could be $d_1^N = 2N - 2$ (in the worst case). The departure time of c_1^N from an (isolated) buffer with service bandwidth exactly N^{-1} is $1 + N$. So, the difference in departure times is of order N .

C. Nonidling Virtual Clock

We now take $v(t) \equiv t$ so that $F_i^n \equiv \mathcal{F}_i^n$. The nonidling bandwidth scheduling policy with this virtual time (originally proposed by L. Zhang [15]) is called Virtual Clock

(or “VirtualClock”). We now prove a minimum-bandwidth result for Virtual Clock (this proof is inspired by that of Theorem 1 of [12] for PGPS).

Theorem 2: Virtual Clock has a minimum-bandwidth property with parameter $\mu = -1$.

Proof:

Consider an arbitrary cell c_i^n .

Let H denote the largest time $t < \lceil F_i^n \rceil$ such that the processor sharing node (PSN) is idle (i.e., no cell departs the PSN at time H and a cell departs the PSN at every time in the interval $[H + 1, \lceil F_i^n \rceil]$). If no such $t \geq 0$ exists, take $H = -1$.

Let J be the largest time $t < \lceil F_i^n \rceil$ such that a cell departs the PSN with VFT $> \lceil F_i^n \rceil$. If no such $t \geq 0$ exists, take $J = -1$ as well.

To prove the theorem, first consider the case where $a_i^n < \max\{H, J\}$. In this case, $d_i^n < \max\{H, J\} < \lceil F_i^n \rceil$, since c_i^n cannot be queued at time $\max\{H, J\}$ due to the definition of H and J . In addition, we see from the definition of H and J that $a_i^n \neq H$ and $a_i^n \neq J$ because of the cut-through mechanism.

For the case where $a_i^n > \max\{H, J\}$, consider the interval of time $I = [1 + \max\{H, J\}, \lceil F_i^n \rceil - 1]$. We now argue that every cell c departing the PSN during interval I has VFT $\leq \lceil F_i^n \rceil$ and arrival time $a \in I$. Clearly, the VFT of c is $\leq \lceil F_i^n \rceil$ by the definition of J . Note that the number of departure epochs during I is $|I| = \lceil F_i^n \rceil - \max\{H, J\} - 1$.

The maximum number of cells c with VFT $\leq \lceil F_i^n \rceil$ and arrival time $a \in I$ is

$$\leq \sum_{i=1}^N \phi_i |I| = |I|;$$

this maximum number includes c_i^n . Since $|I|$ cells depart the PSN during interval I (a cell departs at every integer time $t \in I$), c_i^n must depart during I or before $\max\{H, J\}$. More precisely, we have shown that if $a_i^n < \max\{H, J\}$ then $d_i^n < \max\{H, J\}$; else if $a_i^n \in I$ then $d_i^n \in I$. ■

The minimum-bandwidth parameter of -1 is due to the fact that the processor sharing node has a cut-through implementation (Equation (4)) but the reference FIFO buffer does not (Equation (2)). The distribution of the idle bandwidth among nonidle buffers by Virtual Clock is currently under study.

D. Idling Virtual Clock

We now describe service operation of a simple idling (not work-conserving) bandwidth scheduling policy and prove that it guarantees minimum bandwidths to individual FIFO buffers. As with Virtual Clock, the virtual time function is $v(t) \equiv t$ so that $F_i^n \equiv \mathcal{F}_i^n$. Cells at the head of the buffers are considered for service at each departure epoch. Let the current time be t . The smallest VFT, say F_i^n , of all head-of-buffer cells is found. If $\lceil F_i^n \rceil \leq t$, then c_i^n is chosen for service (ties are broken as above). On the other hand, if $\lceil F_i^n \rceil > t$ then a best-effort cell is served if

one is available. We call this bandwidth scheduler “Idling Virtual Clock.”

Theorem 3: Idling Virtual Clock has a minimum-bandwidth property with parameter $\mu = N - 1$.

Proof:

Define $I_t^n \in \{0, 1\}$ as the number of cell departures at time t from the n^{th} isolated buffer and let $I_t := I_t^1 + I_t^2 + \dots + I_t^N$. Consider a discrete-time queuing process X described by the following Lindley equation:

$$X_{t+1} = (X_t + I_t - 1)^+, \quad t \geq 0,$$

with $X_0 = 0$. X_t is the number of cells in this idling processor sharing node that are “overdue” at time t ; i.e., total the number of cells (not just head-of-buffer cells) in the node at time t whose VFTs are less than or equal to t .

The theorem is proved if we can show that $X_t \leq N - 1$ for all t . For all integers $j \geq i$ define $S_{i,j} = I_i + I_{i+1} + \dots + I_j$. Using Proposition 7.3, p.80, of [1] we get

$$X_t = \max\{S_{0,t-1} - t, S_{1,t-1} - (t-1), \dots, S_{t-2,t-1} - 2, S_{t-1,t-1} - 1, 0\}. \quad (14)$$

Now, since

$$S_{i,j} = \sum_{n=1}^N \left(\sum_{k=i}^j I_k^n \right) \quad \text{and} \quad \sum_{k=i}^j I_k^n \leq 1 + \phi_n(j-i),$$

we get that

$$S_{i,j} - (j-i+1) \leq \sum_{n=1}^N (\phi_n(j-i) + 1) - (j-i+1) \leq N-1.$$

Thus, by equation (14), $X_t \leq N - 1$ as desired since every term on the right-hand side is likewise bounded above. ■

Consequently, under idling Virtual Clock,

$$\lceil \mathcal{F}_i^n \rceil \leq d_i^n \leq \lceil \mathcal{F}_i^n \rceil + N - 1$$

for all n and i .

The bandwidth provided to best-effort traffic by Idling Virtual Clock is more regular in the sense that best-effort cells will depart sooner under Idling Virtual Clock than under Virtual Clock.

D.1 Favours Congested Buffers

We now describe a bandwidth scheduling policy that uses idle bandwidth to alleviate congestion. As before, head-of-buffer cells are considered for service at each departure epoch. Certain buffers³ are said to be *congested* if their occupancy exceeds a predefined threshold. At the current time t , two cases are taken:

Case 1: No buffers are currently congested.

In this case, the smallest VFT, say F_i^n , of all head-of-buffer cells is found and c_i^n is chosen for service.

³For example, those handling loss-insensitive ABR traffic.

Case 2: At least one buffer is congested.

In this case, the smallest VFT among *congested* buffers, say F_j^m , and the smallest VFT among *all nonempty* buffers, say F_i^n , are found. If $t \geq F_i^n$ and $F_i^n < F_j^m$ then c_i^n is served; otherwise c_j^m is served. Only when all N buffers are idle are best-effort cells served.

Thus, in the absence of congestion, this scheduler behaves like nonidling Virtual Clock. The worst case for buffer n is when it is not congested and at least one other buffer is persistently congested resulting in the situation given by Theorem 3 above. To avoid the worst-case delays given by Theorem 3, FIFO buffers handling connections that are delay-sensitive (e.g., voice) could be treated as though they are always congested by this policy.

D.2 Favouring Congested Buffers then Best-Effort Traffic

We now describe a bandwidth scheduling policy that uses idle bandwidth to alleviate congestion; but best-effort traffic is favoured in the absence of congestion. As before, head-of-buffer cells are considered for service at each departure epoch. At the current time t , three cases are taken: **Case 1:** No buffers are currently congested and the best-effort buffer is empty.

In this case, the smallest VFT, say F_i^n , of all head-of-buffer cells is found and c_i^n is chosen for service.

Case 2: No buffers are currently congested and the best-effort buffer is nonempty.

In this case, the smallest VFT, say F_i^n , of all head-of-buffer cells is found. If $F_i^n \leq t$, then c_i^n is chosen for service. On the other hand, if $F_i^n > t$ then a best-effort cell is served.

Case 3: At least one buffer is congested.

In this case, the smallest VFT among congested buffers, say F_j^m , and the smallest VFT among all nonempty buffers, say F_i^n , are found. If $t \geq F_i^n$ and $F_i^n < F_j^m$ then c_i^n is served; otherwise c_j^m is served.

Thus, in the absence of congestion, this scheduler behaves like idling Virtual Clock. The worst case for a noncongested buffer occurs either when at least one other buffer is persistently congested or best-effort cells are queued, resulting in the same situation given by Theorem 3 above. The idle bandwidth is used to serve best-effort traffic in the absence of congestion.

V. IMPLEMENTATION ISSUES

The main objection to PGPS is that its virtual time is computationally complex at high ATM speeds [8]. Note that Virtual Clock has the same implementation complexity as SCFQ (both virtual time functions are easily computed) and has a more appealing minimum-bandwidth property (Theorem 1 versus Theorem 2).

Consider a processor sharing node that has a “fan-in” from the switch fabric of f : i.e., there are at most f cells arriving to the processor sharing node every unit of time (c^{-1} seconds). When implementing a VFT-based bandwidth scheduler, the following operations must be performed at every unit of time in the worst case: $f + 1$ read/write operations to buffer memory (one cell transmission per unit time), f virtual time computations, f “max” operations

and additions (to compute the VFTs), and the minimum of N VFTs must be found (to determine what cell to serve). At high transmission speeds, this amount of computation may necessitate an implementation wherein these operations are interleaved or “pipelined”. At time m , the number of cells X_m^n in the n^{th} FIFO buffer of a processor sharing node having k levels of pipelining is given by the following recursion:

$$\begin{aligned} X_m^n &= X_{m-1}^n + A_{m-k}^n - D_m^n \quad \text{with} \\ X_{-1}^n &= 0 \end{aligned} \quad (15)$$

where the departure indicators D_m^n are determined by the bandwidth scheduler as above. In these terms, we have considered $k = 0$ (i.e., cut-through) implementations in the previous sections.

Comparing Equation (15) with Equation (4), we see that all cells through a “ k -pipelined” processor sharing node are simply delayed exactly k units of time longer than through a cut-through processor sharing node under the same conditions. Consequently, we arrive at the following simple theorem.

Theorem 4: For all $k \in \mathbb{Z}^+$, If a bandwidth scheduler for a cut-through processor sharing node implementation has a minimum-bandwidth property with parameter μ , then that bandwidth scheduler for a k -pipelined implementation has a minimum-bandwidth property with parameter $\mu + k$.

VI. SUMMARY

In summary, we have described a class of bandwidth scheduling policies using virtual finishing times that are suitable for public, wide-area ATM-based B-ISDN. Design goals for such strategies were specified, in particular, the minimum-bandwidth property. PGPS was briefly discussed and a minimum-bandwidth property for SCFQ was proved. Finally, bandwidth scheduling policies using virtual time $v(t) = t$ (including Virtual Clock) were described and minimum-bandwidth results were obtained. The results were obtained under both cut-through and non-cut-through implementations.

The minimum-bandwidth property has been extended recently to tandem processor sharing nodes, see [6] for delay bounds of tandem nodes using Virtual Clock and [9] for a more general setting. That is, end-to-end delay bounds (and buffer sizing results) for arbitrary, tandem processor sharing nodes are obtained in terms of the delay through a reference FIFO buffer with devoted server. These results can be used in the following way to do provisioning for prerecorded video transmission across ATM networks [10]. For a particular connection, the bandwidth requirements are found for an “ideal” network consisting of a single (reference) FIFO buffer; this bandwidth amount *characterizes* the connection and is used as one of its traffic descriptors. The end-to-end results are then used to determine the required bandwidth, buffers, and playback buffer parameters for lossless transmission across an arbitrary virtual channel connection.

REFERENCES

- [1] S. Asmussen. *Applied probability and queues*. Wiley, Chichester West Sussex, 1987.
- [2] R. Cruz. Quality of service guarantees in virtual circuit switched networks. *IEEE JSAC*, Vol. 13, No. 6:pages 1048–1056, Aug. 1995.
- [3] M. de Prycker. *Asynchronous Transfer Mode: Solutions for Broadband ISDN*. Prentice Hall, Englewood Cliffs, NJ, 1991.
- [4] G. de Veciana and G. Kesidis. Bandwidth allocation for multiple qualities of service using generalized processor sharing. Technical Report No. SCC-94-01, U.T. Austin, ECE Dept., 1994, to appear in *IEEE Trans. Info. Th.*
- [5] A. Demers, S. Keshav, and S. Shenker. Analysis and simulation of a fair queueing algorithm. *Internet Res. and Exper.*, Vol. 1, 1990.
- [6] N.R. Figueira and J. Pasquale. An upper bound on delay for the VirtualClock service discipline. *IEEE/ACM Trans. Networking*, Vol. 3, No. 4:pages 399–408, Aug. 1995.
- [7] The ATM Forum. *ATM User-Network Interface Specification Version 3.0*. Prentice-Hall, Englewood Cliffs, NJ, 1993.
- [8] S.J. Golestani. A self-clocked fair queueing scheme for broadband applications. In *IEEE INFOCOM Proc.*, pages 636–646, 1994.
- [9] A. Hung and G. Kesidis. End-to-end delay bounds and buffer sizing in ATM networks. Technical Report 95-08, E&CE Dept, Univ. of Waterloo, June 1995.
- [10] A. Hung and G. Kesidis. Resource management of prerecorded VBR video sources in ATM networks. Technical Report 95-05, E&CE Dept, Univ. of Waterloo, June 1995.
- [11] A.K. Parekh and R.G. Gallager. A generalized processor sharing approach to flow control in integrated services networks: The multiple node case. *IEEE/ACM Trans. Networking*, Vol. 2, No. 2:pages 137–150, Apr. 1994.
- [12] A.K. Parekh and R.G. Gallager. A generalized processor sharing approach to flow control in integrated services networks: The single node case. *IEEE/ACM Trans. Networking*, Vol. 1, No. 3:pages 344–357, Jun. 1993.
- [13] J.W. Roberts. Virtual spacing for flexible traffic control. *International Journal of Communication Systems*, Vol. 7:pages 307–318, 1994.
- [14] G.G. Xie and S.S. Lam. Delay guarantee of virtual clock server. Technical Report TR-94-24, CS Dept, U.T. Austin, Oct. 1994.
- [15] L. Zhang. VirtualClock: A new traffic control algorithm for packet-switched networks. *ACM. Trans. Comp. Sys.*, Vol. 9, No. 2:pp. 101–124, May 1991.

Anthony Hung (S'95) received the B.A.Sc. degree in Computer Engineering from the University of Waterloo, Waterloo, Canada, in 1994. He is currently pursuing a Ph.D. degree, also from Waterloo. His research interests include bandwidth-scheduling and providing quality of service in ATM networks. Mr. Hung is currently working on ATM over satellite with Spar Aerospace, Montreal, Canada, His email address is: ahung@odysseus.uwaterloo.ca

George Kesidis (S'91-M'92) was born in Toronto, Canada, in 1964. He received a B.A.Sc. degree from the University of Waterloo in 1988 and the M.S. and Ph.D. degrees from the University of California at Berkeley in 1990 and 1992, respectively, all in electrical engineering. He is currently an assistant professor in the electrical and computer engineering department of the University of Waterloo, Waterloo, Ontario. His research interests include resource allocation, congestion control and performance evaluation of high-speed networks. His email address is: g.kesidis@eandce.uwaterloo.ca His web site is: <http://cheetah.vlsi.uwaterloo.ca>

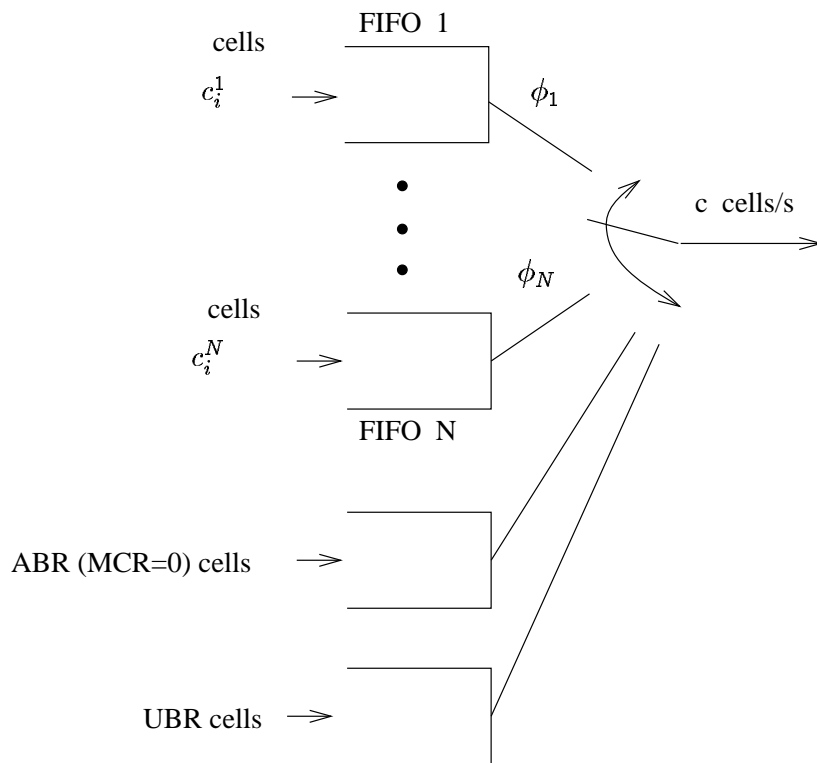


Fig. 1. A Processor Sharing Node