The Ready-to-Go Virtual Circuit Protocol: A Loss-Free Protocol for Multigigabit Networks Using FIFO Buffers

Emmanouel A. Varvarigos, *Member, IEEE*, and Vishal Sharma, *Student Member, IEEE*

***Abstract*—The *Ready-to-Go Virtual Circuit* protocol (or RGVC) is an *immediate transmission* protocol, in which the source need not wait for an end-to-end roundtrip delay for reservations to be made before transmitting the data. The protocol is designed to handle the lossless transport of ABR traffic, and will be used in the 40 Gb/s Thunder and Lightning testbed being prototyped at the University of California at Santa Barbara (UCSB). An important advantage of the RGVC protocol over previous connection and flow control protocols is that it is suitable for networks in which the switches use FIFO buffers that are shared by multiple sessions. The RGVC protocol ensures lossless communication by coupling link capacity with buffer space, so that when a portion of a buffer at a node is occupied, a proportional fraction of the incoming capacity to that buffer is frozen. Given the constraints on the frozen capacity, an algorithm is executed at each node to allocate the transmission rate to each FIFO buffer so as to maximize capacity utilization. The requirement that the protocol operate with FIFO buffers at the network nodes poses some unique challenges in the design that are not present in rate- and credit-based schemes. Briefly, since several sessions share a common FIFO buffer, per-VC flow control is no longer possible so control over the rate of an individual session is lost. Also, since the contents of the buffers change dynamically, the buffer composition becomes difficult to determine. For the rate-allocation algorithm of the RGVC protocol to be executed, however, the contents of the FIFO buffers at a node must be known. To implement the bookkeeping required, we present two schemes: the *measurement-based scheme*, where the bookkeeping function is implemented via measurements, done essentially in hardware; and the *estimation-based scheme*, where the bookkeeping is done analytically via the exchange of control packets between nodes.**

***Index Terms*— FIFO buffers, flow control protocols, switch design.**

I. Introduction

The Thunder and Lightning network (see, for instance, [4], [5], [24], [25]), is a very high-speed, fiber-optic, ATM-based communication network being designed and built at UCSB under the sponsorship of DARPA [10], which will operate at link speeds of up to 40 Gb/s [17]. Our objectives in designing the connection and flow control algorithms for this network are to ensure lossless transmission, efficient utilization of capacity, small pretransmission delay for delay-sensitive traffic, and packet arrival in the correct order. To meet these objectives, we have proposed the Efficient Reservation Virtual Circuit protocol (or ERVC) for constant-rate sessions or for sessions whose rate has some particular smoothness properties, and the Ready-to-Go Virtual Circuit protocol (or RGVC) for delay-sensitive sessions or for sessions whose rate changes with time in an arbitrary way. The ERVC protocol, described in [24], uses reservations and requires little buffering, while the RGVC protocol, which is the subject of the present paper, uses back-pressure and requires buffering at intermediate nodes.

To address the diversity in traffic types in ATM-based networks, the ATM Forum has defined a family of five *service classes*, called the Constant Bit Rate (CBR), the real-time Variable Bit Rate (rt-VBR), the nonreal-time Variable Bit Rate (nrt-VBR), Unspecified Bit Rate (UBR), and the Available Bit Rate (ABR) classes [1]. The ABR service is intended for the economical transport of traffic that does not require firm guarantees on bandwidth and delay, but instead can be sent at a rate that is convenient for the network. The RGVC protocol has been designed with the ABR service in mind.

To support the lossless transport of ABR traffic, a mechanism is needed to handle congestion. The two main classes of flow-control schemes that have been proposed for ABR traffic are the *rate-based* schemes (see, for example, [2], [9], [16], and [23]) and the *credit-based* schemes (see, for example, [11], [14], and [15]). In rate-based schemes, the network sends appropriate information to the sources, specifying the bit-rate at which the sources should transmit, and the feedback control-loop may extend end-to-end across the entire network. The rate-based approach, while inexpensive in terms of implementation complexity and hardware cost, does not handle bursty traffic well. In credit-based schemes, the receiving node sends information to the sending node about the available buffer space and does so independently on a link-by-link basis. While the credit-based approach is well-suited to handle bursty traffic, it requires *per session* (per VC) queueing at the nodes. The need of per session buffering or accounting limits the flexibility of the designer, and is one of the main reasons the ATM Forum selected rate-based schemes for ABR traffic in ATM networks [18], [21]. The high transmission speeds in the Thunder and Lightning network, however, impose the

Manuscript received January 29, 1996; revised May 27, 1997; approved by IEEE/ACM Transactions on Networking Editor C. Partridge. This work was supported by ARPA under Contract DABT63-93-C-0039.

E. A. Varvarigos is with the Department of Electrical and Computer Engineering, University of California, Santa Barbara, CA 93106-9560 USA (e-mail: manos@xanadu.ece.ucsb. edu).

V. Sharma was with the Data Transmission and Networking Laboratory, Department of Electrical and Computer Engineering, University of California, Santa Barbara, CA 93106-9560 USA. He is now with Qualcomm, Inc., San Diego, CA 92121 USA (e-mail: vishal@spetses.ece.ucsb.edu).

Publisher Item Identifier S 1063-6692(97)06547-3.

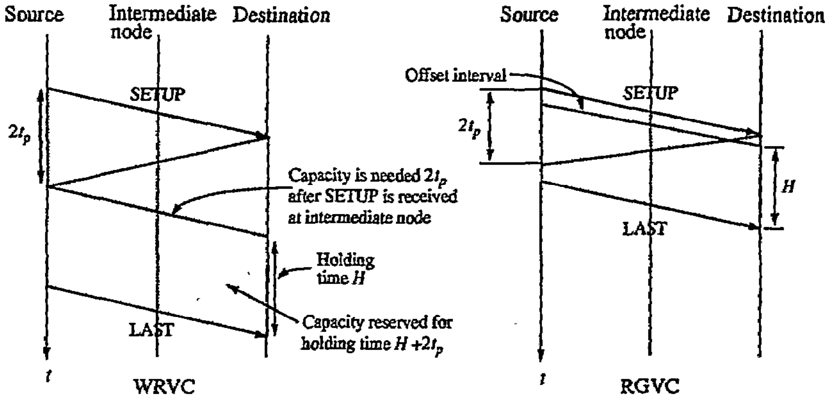

Fig. 1. We illustrate the advantages of immediate transmission protocols, such as the RGVC protocol, over wait-for reservation (WRVC) protocols, for the case when the setup packet is successful in making appropriate reservations.

constraint that an FIFO queueing discipline be used for all packets (including packets belonging to different sessions); as we will see, this is not consistent with the credit-based protocols proposed to date. A major challenge for network research is therefore to design protocols that combine the hardware simplicity of rate-based flow control with the burst handling capability of credit-based flow control. In what follows, it will be seen that, given the architectural constraints of the switch structure (which arise from the extremely high speed at which the Thunder and Lightning network will operate), the RGVC protocol combines many useful features of both rate-based and credit-based flow control schemes. The protocol reacts to congestion faster and can handle bursty traffic better than rate-based schemes, without requiring per-session buffering as credit-based schemes do.

The RGVC protocol will be employed to establish the connection for sessions that cannot tolerate the roundtrip propagation delay required by wait-for-reservation protocols (such as the ERVC protocol) for call setup. In the RGVC protocol, a setup packet is first transmitted over a path toward the destination, followed after a short *offset-interval* by the data packets (see Fig. 1). The data packets are switched based on their virtual circuit identifier (or virtual path identifier) by using the lookup tables established at the intermediate nodes by the setup packet. In this way, a pipelining between the setup phase and the data-transmission phase is achieved, substantially reducing the pretransmission delay. This differs from wait-for-reservation virtual circuit (henceforth called WRVC) protocols, where a pretransmission delay at least equal to one roundtrip propagation delay between the source and the destination is needed before data transmission can begin (this delay can be as large as 45 ms for coast-to-coast communication). This is because, in WRVC protocols, the capacity is blocked for duration equal to at least $H + 2t_p$, where H is the session holding time and t_p is the propagation delay between the source and destination, and the session must wait for the roundtrip delay before beginning transmission. In the RGVC protocol, the session can begin transmission immediately following the offset-interval, and, if the setup packet is successful in reserving the required capacity and congestion does not build up, capacity is only occupied for holding time H plus the duration of the offset interval. In that case, therefore, the RGVC protocol resembles a usual reservation protocol, with the added advantage that the capacity is blocked for other sessions for a much smaller time than in WRVC protocols (see Fig. 1).

If the residual capacity of a link on the path is not adequate, packets may have to be buffered at intermediate nodes, and back-pressure (the details of which we provide in Section IV) is exercised to appropriately control the transmission rates. The RGVC protocol guarantees lossless communication by coupling link capacity with buffer space. In particular, when a portion of the buffer space at a node is occupied, a proportional fraction of the capacity incoming to that buffer is *frozen*, in the sense that it cannot be used by RGVC sessions coming into that buffer. Similarly, when the buffer occupancy decreases, a portion of that incoming capacity is *defrozen*, and is once again available for use by RGVC sessions routed through that buffer. A difference between the RGVC protocol and credit-based schemes is that in the former control packets are sent only when the occupancy of a buffer changes significantly, while in the latter they are sent each time a given number of packets is transmitted from a buffer. This combined with the buffer partitioning scheme, described in Section IV, results in considerably smaller processing and bandwidth requirements for control packets than in credit-based schemes.

The RGVC protocol can operate either with RAM buffers or with FIFO buffers at the network nodes. With RAM buffers, per session (VC) queueing, similar to that used in credit-based schemes, can be exercised. A separate logical queue can be maintained for each session, so that the rate of a particular session can be controlled without affecting other sessions sharing the same buffer. With FIFO buffers, the situation is considerably more complex. Since packets of an individual session cannot be isolated from packets of the other sessions that share the same buffer, it is not possible to control the rate of a particular session without affecting the rates of other sessions. Thus, the actions that a node takes upon the receipt of a control packet differ depending on the nature of the buffers, FIFO, or RAM, that are used at the nodes. FIFO buffers require a more complex back-pressure mechanism than RAM buffers do. This, however, is the price we pay for the much simpler FIFO buffer implementation in very high-speed networks like the Thunder and Lightning network, where link speeds of tens of gigabits per second render RAM buffers infeasible (see also the discussion in Section III). Even though the RGVC protocol can be used with both the FIFO and the RAM buffer implementations, in this paper, we will emphasize the FIFO case, because it poses additional challenges, and it is the one used in the Thunder and Lightning network (for a treatment of the RAM case, we refer the reader to [25]). The requirement of an FIFO queueing discipline at the switches (where the FIFO queue is shared by packets of different sessions) has, to the best of our knowledge, been considered by very few works in the literature (see [20] for an implementation in the Autonet local area network, and [3], [13], and [26] for some related loss-probability analyses), and is one of the main contributions of this paper.

The rate at which the FIFO buffers feeding a particular link should be served to optimize capacity utilization is a nontrivial problem. In our implementation for the Thunder and Lightning network, each node uses a rate-allocation algorithm to maximize the total outgoing rate from a switch given the constraints on the frozen capacity. The execution of the algorithm requires information about the contents of the buffers at the switches. Specifically, the algorithm must know the fraction of data from each FIFO at the sending node that is headed for each of the FIFO's at the receiving node. In our scheme, this information is provided by keeping track of the *FIFO occupancy profile* associated with an FIFO k at the sending node, which records, as a function of buffer depth, the number of packets stored in that FIFO that are destined for each FIFO m at the receiving node. Since the number of such packets varies both with time and as a function of the buffer depth, the FIFO occupancy profile stores a piecewise approximation to the buffer composition with respect to depth, by recording the composition in quantized steps of size M packets (where M is a parameter). The data structure used is a list that for every M packets in the buffer, records how many packets are intended for each FIFO m at the next node.

We propose two schemes to implement the bookkeeping required by our protocol: a *measurement-based scheme*, in which the bookkeeping function is implemented via measurements, done essentially in hardware; and an *estimation-based scheme*, in which the bookkeeping is done analytically using control packets exchanged between nodes. In the measurement-based scheme, a set of hardware counters, which is incremented upon the arrival of a data packet, records the number of packets headed from an FIFO at the sending node to each FIFO at the receiving node. In the estimation-based scheme, the rate allocation algorithm specifies the rates at which the FIFO's at a sending node should transmit in a given interval, which in turn determines the amount of data that will be transmitted to the next node during that interval. Information corresponding to these data is then pruned from the FIFO occupancy list at the sending node, and the pruned information, which is basically a set of numbers, instead of being discarded is transmitted to the receiving node to enable it to build its own FIFO occupancy list.

Note that the rate-allocation algorithm determines the rate at which data *already* accepted in the network should be transmitted from each FIFO, so as to efficiently utilize the capacity of the outgoing links while guaranteeing lossless communication. Even though the RGVC protocol provides flow control within the network to meet its objectives, it allows for considerable flexibility in the way the rate allocated to each source is decided. The rates of the sources can be determined, for example, by a rate-based scheme, like the ones proposed by Siu and Tzeng [21] or by Jain *et al.* ([8] and [19], for instance), that is superimposed on the RGVC protocol to meet the fairness objectives and throughput requirements of the session. Indeed, in a network like the Thunder and Lightning network, which does not drop packets and uses FIFO buffers, it would be difficult to discriminate between packets belonging to different sessions after they have been accepted into the network.

The remainder of the paper is organized as follows. In Section II we provide a general description of the RGVC protocol, and discuss briefly the role of the main control packets used. In Section III, we explain the switch architecture of the Thunder and Lightning network. In Section IV we discuss the difficulties posed by FIFO buffers, and explain the operation of the protocol with an FIFO buffer implementation. In particular, in Section IV-D1 we discuss the measurement-based scheme, and in Section IV-D2 we discuss the estimation-based scheme. Concluding remarks follow in Section V.

II. Overview of the RGVC Protocol

In our description of the protocol, we do not consider issues related to error control and retransmission, since in the Thunder and Lightning network these functions are performed at the transport layer. We also postpone any discussion of issues related to fairness until Section IV-C.

In the RGVC protocol, a SETUP packet is transmitted first over the path to reserve the required capacity and set the routing tables, and is followed after an offset-interval by the data packets. Once a setup packet is processed at a switch and makes the needed reservations, the data packets can be routed through the switch with minimal processing delay, based on their virtual circuit identifier (VCI) or virtual path identifier (VPI). The offset-interval is therefore the minimum time by which the start of the connection-setup phase and the start of the data-transmission phase must be separated to ensure that the data packets do not overpass the setup packet, and it is equal to the number of hops on the path times the difference between the processing times of a setup packet and a data packet. In the remainder of the paper, we will use the terms *sending* (or *upstream*) *node* and *receiving* (or *downstream*) *node* to denote the particular role that a node plays in a given context, and we will use the terms *packet* and *cell* (denoting a 53-byte ATM cell) interchangeably.

If upon the arrival of a setup packet at an intermediate link, the capacity available is inadequate to accommodate the new session, packets start to accumulate at the intermediate node. As the buffer at a receiving (downstream) node starts to build up, the node transmits a FREEZE control packet to the sending (upstream) node that causes the sending node to *freeze* capacity proportional to the buffer space taken up at the receiving node. This means that the frozen capacity is not available for use by sessions at the sending node until the buffer space corresponding to it at the receiving node becomes free. (The frozen capacity can still be used to transmit control packets, however.) A sending node *defreezes* all or part of the frozen capacity only when it estimates, based on the DEFREEZE control packets received, that the buffer space occupied at the receiving node has decreased. This process is different for FIFO buffers than it is for RAM buffers, and will become clearer when we describe it in detail for FIFO buffers in Section IV (the description for RAM buffers is given in [25]). Each node has for every incoming link, buffer space equal to at least $2t_pC$, where t_p is the propagation delay on the link, and C is the link capacity. The exact buffer requirements depend on whether the FIFO or the RAM implementation of the protocol is being used. They also depend on the desired tradeoffs between the protocol complexity and the buffer space per node, as discussed in Section IV-B.

In addition to the control packets mentioned previously, two other control packets used by the protocol are the REFRESH packet and the LAST packet. REFRESH packets are transmitted periodically by the source and inform the intermediate nodes on the path that the connection is active. They are similar in concept to the REFRESH packets used in the connection control protocol discussed by Cidon *et al.* [7], and they ensure that each node periodically learns about the status of ongoing sessions, and that it terminates a session when the REFRESH packet does not arrive within the required time. This guarantees that all reserved bandwidth is eventually released, even in the presence of link and node failures [7]. The LAST packet is transmitted by the source after all data packets of the session have been transmitted, and signals that the session has terminated.

III. Switch Architecture

In this section, we give a conceptual overview of the switch architecture of the Thunder and Lightning network, which we will refer to in our description of the protocol in subsequent sections. For a detailed description of the Thunder and Lightning switch, we refer the reader to [5] and [6].

A network switch has k bidirectional ports, each of which corresponds to an incoming and an outgoing link. Each port has a processor, called switch port processor (or SPP), which is responsible for processing the control packets flowing through the outgoing link of that port (see Fig. 2). An outgoing link transmits packets from k buffers, $k-1$ of which, called *data buffers*, are used by packets arriving on the other incoming links and intended for transmission via this link, while the kth buffer, called *control buffer*, is used by control packets. The reason only $k-1$ data buffers are used is because packets are not allowed to loop back to the node from which they came. The switching hardware handles the movement of the data packets through the switch without involving the SPP. The control buffer has priority over the data buffers, so that the control packets are transmitted without being affected by the data packets. The routing tag (or header) of an incoming packet addresses the routing memory (set by the setup packets of the sessions), which outputs the new routing tag and the link selector bits (see Fig. 2). Each link selector bit corresponds to one of the $k-1$ remaining ports, and is set for each outgoing port for which the packet is intended (required, for example, in multicast operations). The processor bus enables a processor to receive control packets from the incoming ports. If a node is a source for sessions using a port j, one of the data buffers of port j is connected to it. A data buffer n at node i is denoted by $Q_i(n)$. We use the notation $Q_i(S)$ to represent both the buffer used by a session S at node i and the set of sessions that share that buffer, and the notation $|Q_i(S)|$ to denote the buffer space occupied at $Q_i(S)$.

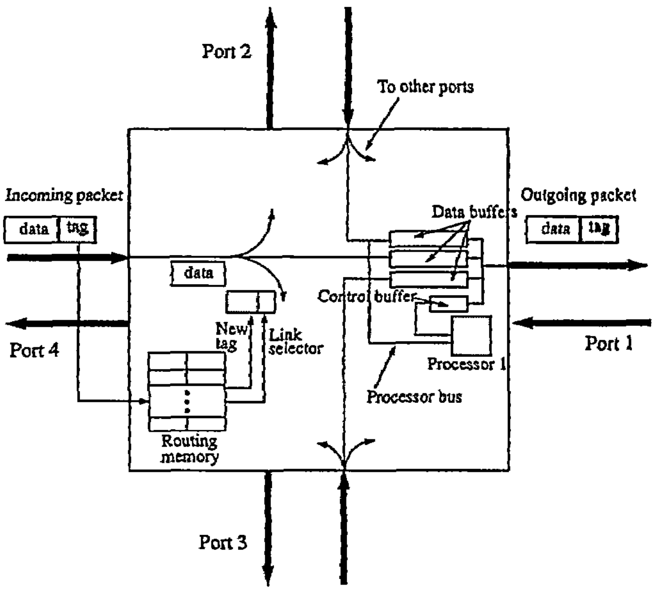

Fig. 2. The architecture of the Thunder and Lightning switch with $k = 4$ ports. (Only the details of Port 1 are shown.).

A major limitation of present day electronic switching, when used in optical fiber networks operating at link rates of tens of gigabits per second, is that the electronics (processors and buffers) is pushed almost to the limit and operates about two orders of magnitude slower than the link rates. For instance, at link speeds of 40 Gb/s, which is the targeted speed of the Thunder and Lightning network, a packet arrives at the switch every 10.6 ns. This corresponds to a packet-arrival rate of 100 MHz, and places severe constraints on the architecture of the switch [4]. The very short time intervals available to perform flow control and session management operations, which require memory accesses and the manipulation of lists or similar data structures, makes per-VPI (or per-VCI) flow control impractical at such speeds [18]. A further issue is that due to the large time-bandwidth product of high-speed networks, even the minimum buffer space needed at a node is large. (For instance, with an interswitch spacing of just 50 km, only one round trip delay worth of ATM cells, the minimum required to ensure lossless communication, translates to about 50 000 cells of storage in the Thunder and Lightning network.) The need to ensure fast access and at the same time maximize chip density and minimize power dissipation, therefore dictates that CMOS buffers be used. (Other alternatives, such as GaAs or ECL, have significantly lower densities and a much higher power dissipation, resulting in nontrivial design and packaging problems.) While CMOS FIFO buffers of considerable size can be designed to keep up with very high transmission speeds, like those of the Thunder and Lightning network [commercially available CMOS FIFO buffers operating at 200 MHz together with half-packet wide (212 bits wide) internal switch paths can yield link speeds of slightly more than 40 Gb/s], with current technology this is not possible to achieve with CMOS RAM buffers, which are needed to implement per-session flow control. In addition to the technological difficulties that it poses, per-session queueing, assumed by most current hop-by-hop flow-control protocols, also poses excessive constraints on network equipment companies, who would like to have more flexibility when designing their systems.

IV. RGVC Protocol with FIFO Buffers

In this section, we discuss a scheme for the operation of the RGVC protocol for very high speed networks like the Thunder and Lightning network, where the nodes must use FIFO buffers. Since a node can no longer isolate the packets of a particular session from those of other sessions sharing the same FIFO, the transmission rate of the entire FIFO through which the session is routed has to be regulated, so that other sessions sharing the FIFO are also affected, at least temporarily. In particular, in Section IV-A we explain the buffer organization and discuss the mechanism for freezing and defreezing of capacity, and in Section IV-B we evaluate the buffer space required to ensure lossless transmission and efficient utilization of the links. In Section IV-C we present the rate allocation algorithm executed at a switch, and in Section IV-D we present two schemes for evaluating the parameters required by the rate-allocation algorithm. In particular, in Section IV-D1) we discuss a measurement-based scheme, while in Section IV-D2) we discuss an estimation-based scheme.

A. Buffer Organization and Freezing of Capacity

The relationship between the buffers at adjacent sending and receiving nodes is illustrated in Fig. 3, where we also illustrate how a single buffer is organized. The organization of an FIFO buffer is based on the concept of *buffer partitioning* to ensure that the number of control packets transmitted by each node is small, and to ensure that short-term fluctuations in the rates of the sessions are smoothed out so that the network need not respond to every change in the rate of a session. Each FIFO buffer with size given by $B = CT$ is partitioned into $K + 1$ bins, with sizes given by B_K, $B_0 = C\tau_0$, and $B_j = C\tau$, for $j = 1, 2, \cdots, K-1$ (see Fig. 3), where C is the capacity of the link, and T, τ_0, and τ are all constants to be determined later. (The constants τ_0, τ, and T can be viewed as the times required for the corresponding bins to fill at the full rate.)

The key idea is that when congestion arises, the input rate to an FIFO buffer is incrementally throttled to prevent buffer overflow. Flow control starts at an output buffer $Q_i(n)$ of a downstream node s_i when the buffer space occupied at $Q_i(n)$ exceeds $B_0 = C\tau_0$. As buffer occupancy rises and crosses a bin boundary B_j (the boundary between bins B_j and B_{j+1}), a FREEZE packet is sent to the upstream node s_{i-1}, asking it to reduce its output rate to $Q_i(n)$ by C/K. The capacity C/K is temporarily *frozen* for $Q_i(n)$, by which we mean that it is not available for use by the sessions routed through $Q_i(n)$. Similarly, as buffer occupancy falls, each time that it crosses a bin boundary B_j, a DEFREEZE packet is sent to the preceding node, which informs the node that it can increase its output rate to $Q_i(n)$ by C/K. The capacity C/K is now *defrozen*, and is once again available for use by new and ongoing sessions routed through $Q_i(n)$. Note that the throttling process depends only on the level of buffer occupancy, and is independent of the particular way in which the input rates to the FIFO buffer change. In particular, node s_i sends at most K successive FREEZE packets when buffer occupancy rises, and at most K successive DEFREEZE packets when buffer occupancy falls.

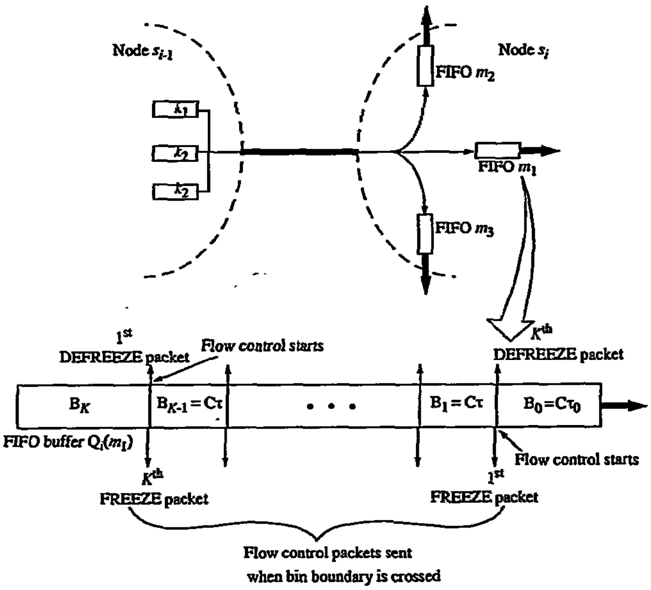

Fig. 3. Each outgoing link in the Thunder and Lightning network is fed by three FIFO data buffers, and in turn feeds into three FIFO data buffers at the receiving node, each corresponding to one of the three outgoing links. Also illustrated is the organization of buffer m_1 at node s_i.

Notice that if the buffer occupancy fluctuates along a bin boundary, the protocol could generate a large number of control packets. This can be corrected by not sending two *successive* control packets for the same boundary. In particular, when buffer occupancy rises and crosses a bin boundary B_j, a FREEZE packet is sent to the preceding node, but when buffer occupancy falls, a DEFREEZE packet is sent to the preceding node only when the occupancy crosses bin boundary B_{j-1}. Note that in this way, capacity C/K may occasionally remain frozen, while it would be safe to use it; if the number of partitions K is reasonably large, the inefficiency created is small. (Fluctuations around the bin boundary B_0 are treated slightly differently, with the capacity being released when the occupancy of B_0 falls below a certain percentage, say, 50%, of the bin capacity.) This mechanism of freezing capacity gradually ensures that at most one control packet is sent per bin's worth of packets even when buffer occupancy fluctuates around a bin boundary, thus reducing the bandwidth and processing expended on control packets. By contrast, if ON–OFF switching was used (for an analysis of the ON–OFF scheme under nonnegligible propagation delay see [26]), a

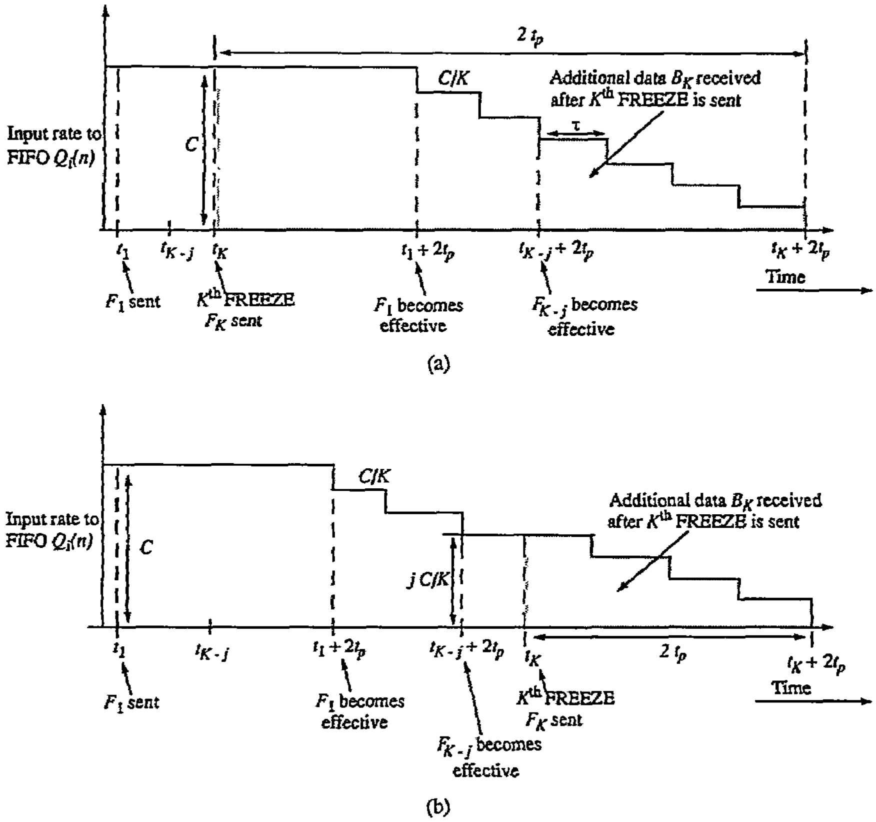

Fig. 4. Illustrates how the allowable (unfrozen) input rate into $Q_i(n)$ changes with time under two scenarios. The input rate into $Q_i(n)$ reduces to zero at time $2t_p$ ms after the Kth FREEZE packet is sent. The shaded area represents the data that will be buffered in the last bin.

large number of control packets could be generated due to fluctuations around the threshold. Another reason for freezing capacity gradually, which we discuss in Section IV-C, is that if the incoming capacity to a buffer is frozen in one step, the upstream FIFO's will all be throttled even though they may primarily have packets for FIFO's other than the one that is congested.

B. Buffer Requirements

We first derive conditions on the parameters τ_0 and τ that ensure that the communication is lossless. To do so, consider a time t_1 at which data arrive at $Q_i(n)$ at the (maximum) rate of C bits/s, and no outgoing capacity is allocated to $Q_i(n)$. Assume also that at time t_1, bin B_0 is full and bins $B_1, B_2, \cdots, B_K$ are empty, so that the first FREEZE packet is sent from $Q_i(n)$. It takes time equal to a roundtrip delay $2t_p$ for the rate reduction requested by a FREEZE packet to become effective at node s_i. Buffer $Q_i(n)$ must therefore have space to store the packets that continue to arrive at node s_i between time t_1 and the time at which the input to s_i ceases completely.

Let t_K be the time at which the Kth FREEZE packet is sent. Observe that at time t_K, bins $B_0, B_1, \cdots, B_{K-1}$ at $Q_i(n)$ are already full, so that the buffer space occupied at that time is equal to $B_0 + (K-1)C\tau$. To obtain conditions for lossless communication, consider the two possible scenarios illustrated in Fig. 4.

In scenario A [see Fig. 4(a)], all FREEZE packets, $F_1, F_2, \cdots, F_K$, are transmitted before the first of them, F_1, becomes effective at $Q_i(n)$. The additional buffer space B_K, needed to store the data that arrive after time t_K and before time $t_K + 2t_p$ (when the inflow ceases completely), is given by the shaded area in Fig. 4(a) as

$$B_K \leq 2Ct_p - \frac{C(K-1)\tau}{2}. \tag{1}$$

In scenario B [see Fig. 4(b)], a total of $K-j$ FREEZE packets, $F_1, F_2, \cdots, F_{K-j}$, have already become effective at $Q_i(n)$ by the time t_K at which F_K is sent, and the incoming rate is at most equal to jC/K. In the time interval between t_K (at which F_K is sent) and $t_K + 2t_p$ (at which F_K becomes effective), a total of j FREEZE packets become effective at $Q_i(n)$ resulting in corresponding drops in the available (unfrozen) capacity. The amount of data arriving at $Q_i(n)$ is maximized when the j drops (of magnitude C/K) in the available capacity occur in succession at the very end, at intervals of τ ms. Therefore, the worst case additional buffer space B_K is given by

$$B_K \leq \frac{C}{K}\left[2t_p j - \frac{j(j-1)\tau}{2}\right]. \tag{2}$$

The right-hand side of (2) is maximized when

$$j = \text{Rnd}\left(\frac{2t_p}{\tau} + \frac{1}{2}\right) \tag{3}$$

where Rnd(x) represents the integer closest to x. Relaxing the constraint that j be an integer (and therefore obtaining an

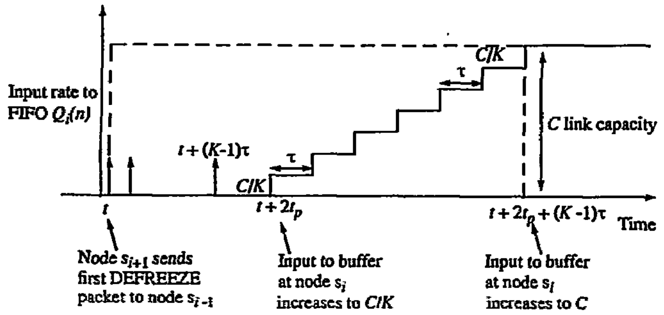
Input rate to FIFO $Q_i(n)$
$t+(K-1)\tau$
τ
C/K
τ
C/K
C link capacity
t
$t+2t_p$
$t+2t_p+(K-1)\tau$
Time
Node s_{i+1} sends first DEFREEZE packet to node s_{i-1}
Input to buffer at node s_i increases to C/K
Input to buffer at node s_i increases to C

Fig. 5. Illustrates how the *defreeze* procedure works in a scenario used to obtain conditions on the buffer parameters to enable efficient utilization of the links.

upper bound on B_K), we obtain

$$B_K \le \frac{C}{K}\left[\frac{2t_p^2}{\tau} + t_p + \frac{\tau}{8}\right]. \qquad (4)$$

Since in scenario B we have $j \le K-1$, (3) also gives

$$2t_p \le (K-1.5)\tau. \qquad (5)$$

Substituting for t_p in the expression for B_K above, we get

$$B_K \le \frac{C(K-1)\tau}{2} - \frac{C\tau}{2}\left(1-\frac{1}{K}\right) \le \frac{C(K-1)\tau}{2}.$$

Therefore, the total buffer space B needed at $Q_i(n)$ to ensure lossless communication (irrespective of the times at which the FREEZE packets are sent) is given by

$$\begin{aligned} B &= B_0 + C(K-1)\tau \\ &\quad + \max\left\{2Ct_p - \frac{C(K-1)\tau}{2}, \frac{C(K-1)\tau}{2}\right\}. \end{aligned} \qquad (6)$$

The condition of (6), which ensures lossless communication, provides for considerable flexibility in choosing the various parameters. In specifying the parameters τ_0 and τ, we also want to ensure that link capacity does not remain idle unnecessarily. In particular, if the buffer starts emptying and the incoming capacity starts defreezing, the buffer should have sufficient packets to keep the outgoing link busy until it begins receiving packets on the incoming link. Consider the scenario illustrated in Fig. 5 (assuming the outgoing link also has capacity C), and assume that at time t the bins $B_0, B_1, \cdots, B_{K-1}$ at buffer $Q_i(n)$ are full, the capacity at $Q_i(n)$ is frozen, the bin B_K is empty, and $Q_i(n)$ is granted an output rate of C. Buffer $Q_i(n)$ then starts emptying at the rate of C bits/s, and, as shown above, sends at most K DEFREEZE packets to the preceding node s_{i-1} at τ second intervals. Therefore, the input rate to buffer $Q_i(n)$ becomes equal to C/K at time $t+2t_p$ and increases to C at time $t+2t_p+(K-1)\tau$. For the outgoing link at node s_i to remain continuously busy, we must have that the amount of data transmitted over the outgoing link at node s_i in the interval $[t, t+2t_p+(K-1)\tau]$ is less than the data that it has at time t and the data that it receives in the interval $[t, t+2t_p+(K-1)\tau]$, that is,

$$\begin{aligned} C[2t_p+(K-1)\tau] &\le B_0 + \sum_{i=1}^{K-1} B_i + \frac{C(K-1)\tau}{2} \\ &= C\tau_0 + \frac{3C(K-1)\tau}{2}. \end{aligned}$$

For the above scenario, the outgoing link is thus guaranteed to be busy when τ_0 and τ are chosen so that the condition

$$2t_p \le \tau_0 + \frac{(K-1)\tau}{2} \qquad (7)$$

is satisfied.

The selection of appropriate values for τ_0, τ, and K that satisfy (6) and (7) depends on the desired tradeoffs between implementation complexity and efficiency of operation. For instance, a small value for K would be inefficient, especially during fluctuations of buffer occupancy around a bin boundary, since capacity equal to C/K could remain frozen unnecessarily (see Section IV-A). A very large value for K, on the other hand, would result in too many control packets being generated, and considerable work for the SPP that has to process them. Similarly, a small value for τ_0 could cause capacity to be frozen too quickly, thereby overreacting to short-term fluctuations, whereas a large value for τ_0 would increase the total buffer space required. One possible solution to (7) is to set

$$\tau_0 = t_p, \text{ and } (K-1)\tau = 2t_p \qquad (8)$$

which gives a total buffer space equal to $B = 4t_pC$. In theory, the minimum buffer space required for lossless transmission is $2t_pC$ if an ON–OFF scheme is followed, but would require that the OFF threshold beset at zero packets, and it is about $3t_pC$ if capacity is frozen gradually as described above. In the Thunder and Lightning network switch, each FIFO buffer can hold up to $B = 90 \times 10^3$ packets. Since the link capacity is $C = 40$ Gb/s or 94.34×10^6 packets/s and the propagation delay per unit of fiber length is $d = 5$ μs/km, the spacing L between two successive switches should satisfy $4dLC \le B$, which for the above parameters gives $L \le 50$ km.

We point out that the buffer size in the RGVC protocol is a function only of the roundtrip delay between adjacent nodes, and is independent of the number of sessions carried. In contrast, in credit-based protocols, the total buffer size per node grows linearly with the number of sessions flowing through it, with the constant of proportionality being equal to the parameter N_2, which is the inverse of the rate at which credit update cells are transmitted from the receiving node to the sending node [14] (this may be minimized using group-based buffer reservation [12]). In multigigabit networks, where a large number of sessions maybe sharing a link (in an ATM cell, 12 bits are used for the VPI number, which, if taken literally, provide for as many as 4096 sessions sharing a link), even moderate values for N_2 would result in large buffer requirements.

Furthermore, in the RGVC protocol, no control packets are transmitted when the buffer occupancy does not change significantly, unlike credit-based schemes [14], where a credit-update cell must be sent by the receiving node to the sending node, for each ongoing virtual circuit, after removing a given number N_2 of data cells of a circuit, even if buffer occupancy stays the same. Therefore, flow control in the RGVC protocol

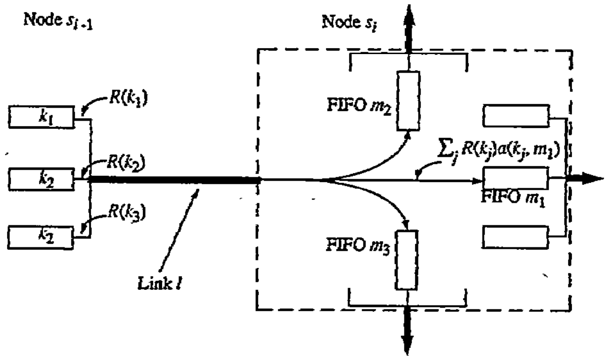

Fig. 6. The data flow between the FIFO buffers at two successive nodes. The total inflow into an FIFO m_l at node s_i is equal to the sum of the rates $R(k_j)$ at which data are emitted by the FIFO's k_j at node s_{i-1}, times the fraction $a(k_j, m_l)$ of data that flows from an FIFO k_j at node s_{i-1} to an FIFO m_l at node s_i.

is "silent," and does not produce any overhead when it is not needed. Note also that credit-based protocols require per session buffering and cannot handle the case where *different* sessions share the same FIFO buffer, while, as we will show, the RGVC protocol is consistent with FIFO buffering. A receiving node needs to send only one control packet per FIFO (as opposed to per session) to the sending node, and needs to do so only when the buffer occupancy changes significantly, resulting in considerably lower processing and bandwidth overhead than in the credit-based schemes. In particular, in our scheme the fraction of capacity taken by control packets is equal to at most $1/C_T$, while in credit-based schemes, the overhead per session is equal to $1/N_2$, which can be as high as 10% of the session's bandwidth [14].

C. Rate Allocation Procedure at a Switch

In the previous subsections we explained how the FIFO buffers in the RGVC protocol are organized, and discussed the requirements on the buffer space to ensure lossless transmission. In this subsection, we discuss how a sending node s_i allocates the transmission rates $R(k)$, $k \in \{k_1, k_2, k_3\}$, to each of the FIFO data buffers feeding an outgoing link L (see Fig. 6).

We define the *occupancy* $S(k)$ at an FIFO k, $k \in \{k_1, k_2, k_3\}$, as 4

$$S(k) = \frac{|Q(k)|}{T_p}, \tag{9}$$

where $|Q(k)|$ is the buffer space occupied at $Q(k)$, and T_p is a parameter that is at least as large as the time between successive executions of the rate allocation algorithm, and is determined by the speed of the switch port processor. The occupancy $S(k)$ can be viewed as the rate at which FIFO k should transmit to clear the occupied buffer space within time T_p.

Our rate allocation algorithm attempts to maximize the total outflow from a node, under the constraint that the sum of output rates $R(k)$, $k \in \{k_1, k_2, k_3\}$, is less than the capacity C of the outgoing link and that the input rate to downstream FIFO's is less than what they can accept without buffer overflow. We let $a(k, m)$ be that fraction of the data output from an FIFO k, $k \in \{k_1, k_2, k_3\}$, at node s_{i-1} that is destined for an FIFO m, $m \in \{m_1, m_2, m_3\}$, at node s_i (see Fig. 6). Note that $\sum_{m=m_1}^{m_3} a(k, m) = 1$ for each k, except when some of the sessions using FIFO k are multicast, in which case $\sum_{m=m_1}^{m_3} a(k, m) > 1$. We assume that the fractions $a(k, m)$, $k \in \{k_1, k_2, k_3\}$, and $m \in \{m_1, m_2, m_3\}$, are known, and we describe in Section IV-D two ways in which they can be evaluated. We also let $F(m)$ be the last estimate that the sending node s_{i-1} has about the frozen input capacity of an FIFO m, $m \in \{m_1, m_2, m_3\}$, at the receiving node s_i. The rate allocation problem is then formulated as the following linear programming problem:

(Problem) P

$$\max \sum_{k=k_1}^{k_3} R(k)$$

subject to

$$\sum_{k=k_1}^{k_3} R(k)a(k, m) \le C - F(m), \quad \text{for } m \in \{m_1, m_2, m_3\} \tag{10a}$$

$$\sum_{k=k_1}^{k_3} R(k) \le C \tag{10b}$$

and

$$R(k) \le S(k), \qquad \text{for } k \in \{k_1, k_2, k_3\} \tag{10c}$$

where $S(k)$ is given by (9). Problem P is solved either when the $F(m)$ or $a(k, m)$ change for some m or k, or at intervals of T_p ms, whichever happens first. Packets are transmitted from FIFO k at rate $R(k)$ until problem P is solved again, which happens after at most T_p ms. Equations (9) and (10c) guarantee that data $R(k)T_p$ is available at the FIFO to transmit at that rate for duration T_p. Solving problem P without the constraint (10c) might result in a rate allocation that maximizes the rate $\sum_{k=k_1}^{k_3} R(k)$ *allowed* for the link, but would not maximize the *actual* transmission rate on the link, which might be smaller due to unavailability of packets. The rate $R(k)$ can be enforced for FIFO k through the use of a permit-based mechanism, with one permit being generated every $1/R(k)$ seconds. (The actual implementation in the Thunder and Lightning network is slightly different, but we will assume that the rate $R(k)$ is constant throughout a transmission interval T_p, since that enables a clearer exposition of the protocol's operation.)

If one of the FIFO's, say FIFO k_1, is a source of a session S, it may be advantageous to give it lowest priority when allocating the rates. This is done by assuming $R(k_1) = 0$, solving problem P only for the two variables $R(k_2)$ and $R(k_3)$ that remain, and then setting

$$R(k_1) = \min_{m \in N_S} \left[C - F(m) - \sum_{j=k_2}^{k_3} R(k)a(k, m) \right] \tag{11}$$

where N_S is the set of FIFO's at node s_i through which session S is routed (N_S may include several FIFO's if a session S is multicast). The reason for doing this is that reducing the rate $R(k_1)$ of a session originating at a link can be done easily and has less severe effects on the network than reducing $R(k_2)$ or $R(k_3)$. This is because FIFO k_1 does not receive packets from other nodes, and, therefore, the transmission of control packets and the freezing of capacity at other nodes is not required when regulating the rate $R(k_1)$ of a source. Recall that our aim is to maximize utilization of our resource, namely, network bandwidth. Thus, throttling the source closest to the point of congestion, instead of the source of the particular session, minimizes the amount of frozen capacity in the network and the number of sessions affected by the throttling action, and maximizes bandwidth usage. Viewed in another way, packets already in the network (which occupy precious resources) are given higher priority than packets that are about to enter the network. Through a separate mechanism, a source that started transmission at a rate higher than what it was finally allocated receives a control packet, which informs it of the rate finally allocated, but packets of the session already in the network are not dropped and are eventually delivered.

An important advantage of the RGVC protocol over threshold-based schemes is that it reacts to congestion in a phased manner. This is particularly significant when several sessions share an FIFO buffer at the sending node but go to different FIFO's at the receiving node (as is the case for our protocol). In the presence of congestion, a threshold-based scheme would freeze the entire link capacity, needlessly penalizing FIFO's that contain sessions destined for FIFO's other than the one that is congested, and possibly creating further unnecessary congestion in the network. Freezing capacity gradually, on the other hand, allows FIFO's not transmitting to the congested one to continue transmission at a (possibly) reduced rate, and, also helps in absorbing short-term fluctuations.

While the RGVC protocol provides for flow control *within* the network to meet the objectives of lossless transmission and efficient link usage, it provides for several choices in the way rates are allocated to the sources at the periphery of a network. To meet the long-term fairness objectives, rate-based schemes, such as the *intelligent congestion control* scheme of Siu and Tzeng [21] or the *ERICA+* scheme of Jain *et al.* [8] can be superimposed on the RGVC protocol. The key idea of these schemes is to periodically advise the sources about the rates at which they should transmit, by monitoring the load at the switches and computing the bandwidth allocation of each source. Each congested switch therefore estimates the "optimal" rate for each VC, using either a first-order filter [21] or a link load-factor and the max-min fairness criterion [8]. This rate is then conveyed to the source using a simple feedback mechanism, and is used by it to adaptively adjust the rates of its sessions. Note that it takes at least one roundtrip delay for the action taken by a source to become effective at the point of congestion. Such a scheme can therefore be used to complement the action of the RGVC protocol. While the RGVC protocol is used to alleviate congestion locally and optimize link usage, a rate-based scheme can be used to ensure that by adjusting source rates congestion is minimized in the long run.

D. Determining Buffer Composition

In this subsection, we discuss how the composition of packets in the buffer is determined. The composition is needed to calculate the fractions $a(k, m)$, $k \in \{k_1, k_2, k_3\}$ and $m \in \{m_1, m_2, m_3\}$, used by the rate-allocation algorithm, which represent the proportion of the data output from FIFO k at sending node s_{i-1} during an interval of length T_p ms that is destined for FIFO m at receiving node s_i. For simplicity, we will refer to a packet that travels from FIFO k to FIFO m as a packet of *class m*. In the Thunder and Lightning network, each outgoing packet at an FIFO may belong to one or more of three classes, depending on whether the session is being *unicast* or *multicast*. The main problem here is that the fractions $a(k, m)$ vary dynamically both with time and as a function of buffer depth. The solution is to record a piecewise approximation to the variation of $a(k, m)$ with respect to time.

In our scheme, a node keeps track of the number of packets of each class m stored in each FIFO. In theory, this information can be completely specified by recording, e.g., as a linked list of records, the type m of each packet located at each FIFO k. In practice, it can be realized by recording the composition of packets only for *quantized blocks* of size M. In particular, in our implementation, each node maintains an *FIFO occupancy list* $\mathcal{F}_k$, for each FIFO k, $k \in \{k_1, k_2, k_3\}$, feeding an output link. Each record of $\mathcal{F}_k$ stores data for one block of packets, and is composed of two fields—the *packet-count field* and the *block-size field*. The packet-count field is an array of three elements, $N_k(m)$, $m \in \{m_1, m_2, m_3\}$, which record the number of packets of each type m in a data block with a size specified by the block-size field. In other words, the jth record $\mathcal{F}_k^j$ of the list stores the numbers $N_k^j(m)$, $m \in \{m_1, m_2, m_3\}$ of packets of class m in the jth block of packets, and the size M_j of that block. In our case, the block-size field is equal to M except, possibly, for the first and the last. The rationale behind recording the composition of arriving packets in terms of blocks of a fixed size M, rather than in terms of blocks of variable size, is to guarantee that the storage overhead is limited to one record per M data packets. (Otherwise, the size of the blocks may get smaller and smaller leading to a greater number of discontinuities in the resulting FIFO occupancy profile, and more processing requirements.) The size M is chosen depending on the processing power and the memory available at the SPP. Fig. 7 illustrates how an FIFO occupancy list may appear when the buffer has L blocks of packets in it.

We discuss the list update procedure next, where we present two schemes for performing the update. In the first scheme, called the *measurement-based scheme*, the list is updated by appropriate measurements of the incoming flow in hardware; whereas in the second scheme, called the *estimation-based scheme*, the list is updated by analytic means and the exchange of control packets between successive nodes.

1) Measurement-Based Scheme: In this scheme, the composition of each block of incoming packets is determined by measurements, which, for very high speed networks, are performed in hardware. In particular, the composition of each block of M packets can be recorded by placing three counters—one each for counting the number of packets of class m_1, m_2, and m_3, respectively, at the input of an FIFO k. Here m_1, m_2, and m_3 correspond to the three FIFO's at the next node to which packets from FIFO k may travel. Each

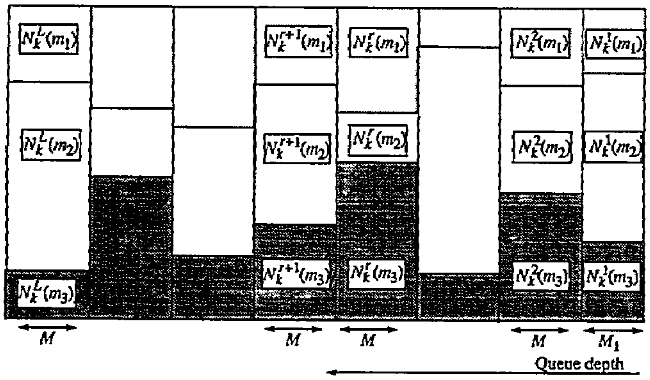

Fig. 7. The *FIFO occupancy list* for an FIFO k. The horizontal axis represents buffer occupancy in terms of blocks of size M, except the first block, which is of size M_1. The vertical axis represents the composition of each block in terms of the number of packets of each class m of which it is composed.

time the sum of the counters becomes equal to M, the values in the three counters are recorded, and the counters are reset. The values in the counters are precisely the numbers $N_k(m_1)$, $N_k(m_2)$, and $N_k(m_3)$ of packets of each class contained in the block of M packets, and are used to create a new record that is added to the end of the FIFO occupancy list for FIFO k (see Fig. 8). To identify the particular combination of FIFO's m_1, m_2, and m_3 at the next node for which a packet is intended, the routing memory contains some additional bits, called the *FIFO selector bits*, which are set by the setup packet during the connection setup phase. The number of bits needed in the routing memory varies from three bits for unicast routing to nine bits for multicast routing. This is because when multicasting is permitted, the packet entering a port at a receiving node may be routed through any combination of its remaining three ports, and could therefore be headed for any combination of the nine FIFO's at the following three nodes, giving a total of 2^9 possible routing combinations.

During the interval between two successive updates, an FIFO k transmits at the rate $R(k)$ that was allocated to it during the previous rate update. In our implementation, the updates happen at regular intervals, called *transmission intervals*, of duration $\Delta t = T_p$ ms. If, however, the rate allocation algorithm is event-driven and is run whenever $a(k, m)$ or $F(m)$ changes, we will have in general $\Delta t \neq T_p$, (but always $\Delta t \leq T_p$). In such a case, Δt can be determined by using an *interval timer*, which is set to zero each time that the FIFO transmission rates are updated via the rate allocation algorithm. In the following, we assume $\Delta t = T_p$, as is the case for the Thunder and Lightning network.

We now describe how the list elements that correspond to packets already transmitted are deleted after a transmission interval of duration Δt. Let r be the largest integer such that

$$\sum_{j=1}^{r} M_j \leq R(k)\Delta t. \tag{12}$$

In other words, r is the number of packet blocks that are completely transmitted from FIFO k in the Δt ms following the previous rate update. To update the list, the first r elements are deleted, and the $r + 1$th element (which is now the first element of the revised list) is modified by updating the block-size field to

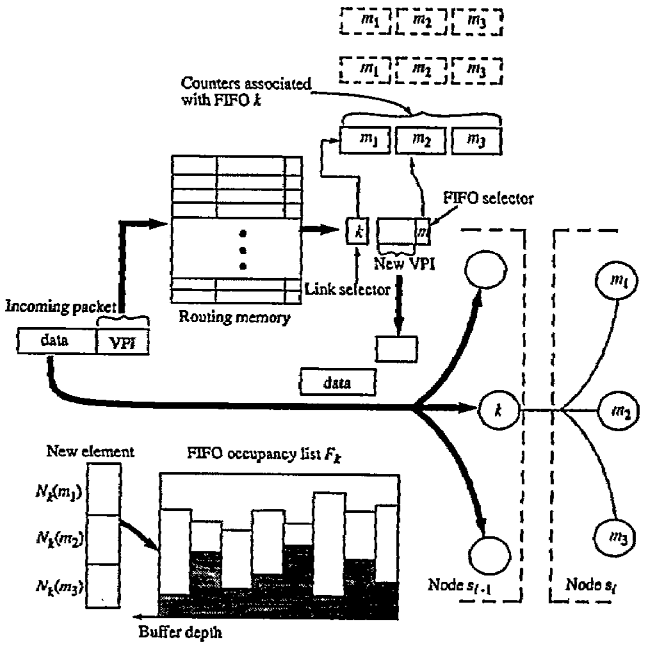

Fig. 8. The layout for obtaining the composition of a block of incoming packets in the measurement-based scheme. We also illustrate the addition of a new element to the FIFO occupancy list of FIFO k.

$$\hat{M}_1 = M_{r+1} - \left[R(k)\Delta t - \sum_{j=1}^{r} M_j\right] \tag{13a}$$

and the packet count field to

$$\hat{N}^1(m) = \frac{N^{r+1}}{M_{r+1}} \cdot \hat{M}_1, \qquad \text{for } m \in \{m_1, m_2, m_3\} \tag{13b}$$

where the hat denotes the values of the elements with new indices (i.e., after the list is updated). In writing (13b) we assumed that the packets of the three classes are distributed uniformly in block $r + 1$. The assumption of uniform relative ratios within a block is an approximation (the distribution depends upon the exact order in which the packets arrive, which is not recorded).

We have seen up to now how the FIFO occupancy list is updated by adding new elements and purging elements that correspond to packets transmitted. We now show how the fractions $a(k, m)$ required by the rate-allocation algorithm are calculated. Observe that the linear programming problem P in Section IV-C specifies the rates $R(k)$ as a function of the fractions $a(k, m)$, or

$$R(k) = F[a(k, m)], \qquad m = m_1, m_2, m_3 \tag{14}$$

while the fractions $a(k, m)$ themselves are found by averaging over the $R(k)T_p$ packets and therefore depend on the rates $R(k)$ allocated, that is,

$$a(k, m) = G[R(k)]. \tag{15}$$

This is because the amount of data sent from an FIFO k to the three FIFO's at the next node and also the fractions of data going to each of the three FIFO's at the next node depends on the rate $R(k)$. Equations (14) and (15) may be solved jointly using a successive approximation algorithm (see the Appendix and Fig. 9) to find the optimal rates $R(k)$.

2) Estimation-Based Scheme: In the scheme described above, the number of packets of type m added to the list was evaluated by measurements in hardware. In that case, only the frozen capacity $F(m)$ at the receiving FIFO's needs to be transmitted to the sending node, and no other control packets are associated with the flow control. The additional hardware needed for hardware measurements (even though simple) may be undesired by a designer, especially for very fast switches, where the layout of the data and control paths is already complicated by the need to make the paths sufficiently wide to reduce the internal speed of the switch [5]. In the following, we present an alternative scheme where this number is estimated by analytic means and the transmission of control packets between nodes. In our presentation, we assume that the rate-allocation algorithm P is run every T_p ms (i.e., $\Delta t = T_p$), and that the rate $R(k)$ at which packets are transmitted from an FIFO k during these T_p ms is constant. To eliminate the need for measurement of the inflow, a receiving node s_i should know: a) the number of packets that a sending node s_{i-1} will transmit to it in a given transmission interval; and b) how many of these packets belong to each class m, $m \in \{m_1, m_2, m_3\}$. (Recall that a packet of class m is headed for FIFO m at a receiving node.) Observe that the information that s_i needs is similar to what s_{i-1} prunes from its occupancy list when the list is purged. Thus, the first requirement is satisfied if the sending node s_{i-1} sends to the receiving node s_i information about the portion of its FIFO occupancy list that it will prune. The second requirement is satisfied if the FIFO occupancy profile for each FIFO is maintained on a per-session basis. This is necessary in the estimation-based scheme, since sessions that share an FIFO k at a sending node may use different FIFO's at the receiving node. Then, by summing over all sessions that are headed for the same FIFO at the next node, node s_i can calculate how many packets in a given block belong to each class m, $m \in \{m_1, m_2, m_3\}$. In the estimation-based scheme, each node keeps track of the FIFO occupancy profile by maintaining an FIFO occupancy list $\mathcal{F}_k$ as in Section IV-D-1. Now, however, each element of the list records the composition of a packet block by recording the number of packets of *each session* $\mathcal{S}$ routed through FIFO k that are present in that block. Thus, the packet-count field is itself a linked list, each element of which records the number $N_k(\mathcal{S})$ of packets of a particular session $\mathcal{S}$ that are present in the corresponding packet block. The data block size of each element in the list (except possibly for the first and the last) is the same, and is equal to M packets.

Consider the situation at a node s_{i-1} just before the start of a new transmission interval, and assume that the FIFO occupancy list has L elements. In order to solve problem P and calculate the transmission rates $R(k)$, $k \in \{k_1, k_2, k_3\}$, for this interval, node s_{i-1} needs to evaluate the fractions $a(k, m)$, $m \in \{m_1, m_2, m_3\}$. These fractions can again be calculated by using the successive approximation algorithm given in the Appendix.

In order to update the occupancy list $\mathcal{F}_k$, the sending node s_{i-1} finds the largest integer r such that

$$\sum_{j=1}^{r} M_j \leq R(k)T_p \tag{16}$$

and updates the fields of the $(r+1)$th element of the list (which will become the first element of the revised list, when the first r elements are purged) according to

$$\hat{M}_1 = M_{r+1} - \left[R(k)T_p - \sum_{j=1}^{r} M_j\right] \tag{17}$$

and

$$\hat{N}^1(\mathcal{S}) = \frac{N^{r+1}(\mathcal{S})}{M_{r+1}} \cdot \hat{M}_1, \qquad \text{for all } \mathcal{S} \text{ in } \mathcal{F}_k^{r+1}. \tag{18}$$

In writing (18) we assumed, as we did in Section IV-A, that the relative ratios of packets of the sessions are uniform throughout the entire block. Once again, the "hat" denotes the values of the elements in the updated list.

The elements that are purged from the occupancy list of the sending node are sent to the downstream node to help it update its own occupancy list. The sending node forms a *transmission list* $\mathcal{T}_k$, which is of the same type with the occupancy list $\mathcal{F}_k$, and contains essentially the portion of the list $\mathcal{F}_k$ that is purged. In particular, the first r elements of $\mathcal{T}_k$ are

$$\mathcal{T}^j = \mathcal{F}_k^j, \qquad \text{for } j = 1, \cdots, r \tag{19}$$

while the packet-count field $\overline{N}$ and the block-size field $\overline{M}$ of the $(r+1)$th element are given by

$$\overline{N}^{r+1}(\mathcal{S}) = \hat{N}^1(\mathcal{S}) - N^{r+1}(\mathcal{S}), \qquad \text{for all sessions } \mathcal{S} \text{ in } \mathcal{F}_k^{r+1} \tag{20a}$$

and

$$\overline{M}^{r+1} = M_{r+1} - \hat{M}_1. \tag{20b}$$

The transmission list $\mathcal{T}_k$ is sent to the receiving node s_i (see Fig. 10), which uses the lists received from the three FIFO's at the sending node s_{i-1} to update the FIFO occupancy lists at its own FIFO's m_1, m_2, and m_3. If the number of sessions routed through an FIFO is equal to N, the overhead for transmitting the lists involves the transmission of at most $2N$ numbers (the VPI and the packet count) for each block of M packets, and can be made small by increasing M.

Fig. 10 illustrates the way the occupancy profile at a particular FIFO, say FIFO m_2, at the receiving node s_i is updated to take into account the new packets that arrive (the case of the remaining two FIFO's is similar). The SPP that controls FIFO m_2 isolates those portions of the transmission lists $\mathcal{T}_k$ that are intended for it. Since the FIFO transmission rates $R(k)$, $k \in \{k_1, k_2, k_3\}$ are constant over the duration T_p of the transmission interval, the $R(k_1)T_p$ packets arrive from FIFO k_1 at a regular rate over a period of T_p ms. The same is true of the $R(k_2)T_p$ packets arriving from FIFO k_2, and for the $R(k_3)T_p$ packets arriving from FIFO k_3. The *arrival profile* from each FIFO k spans a duration of T_p ms, with the

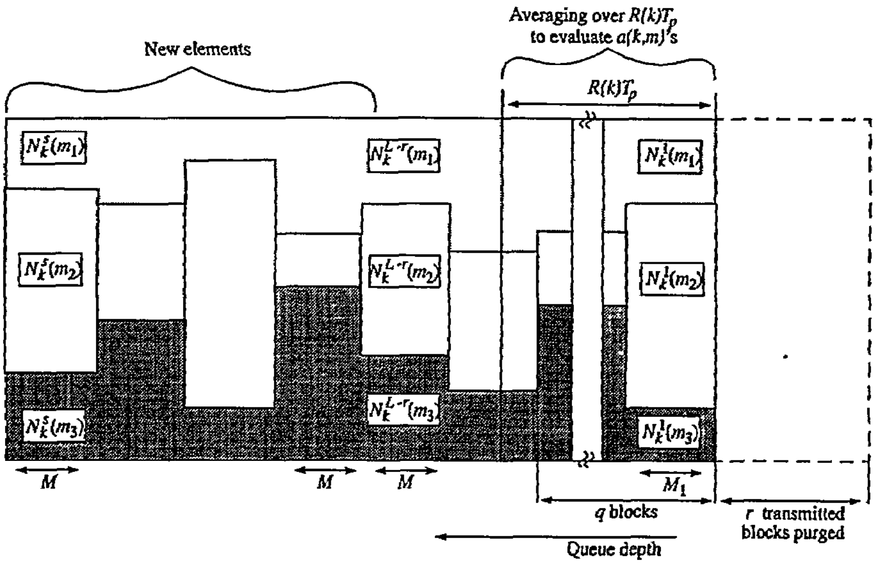

Fig. 9. The calculation of the fractions $a(k, m)$ in the measurement-based scheme. The node averages over the smaller of $R(k)T_p$ packets or the number of packets present in the buffer to obtain an initial estimate for the fractions $a(k, m)$.

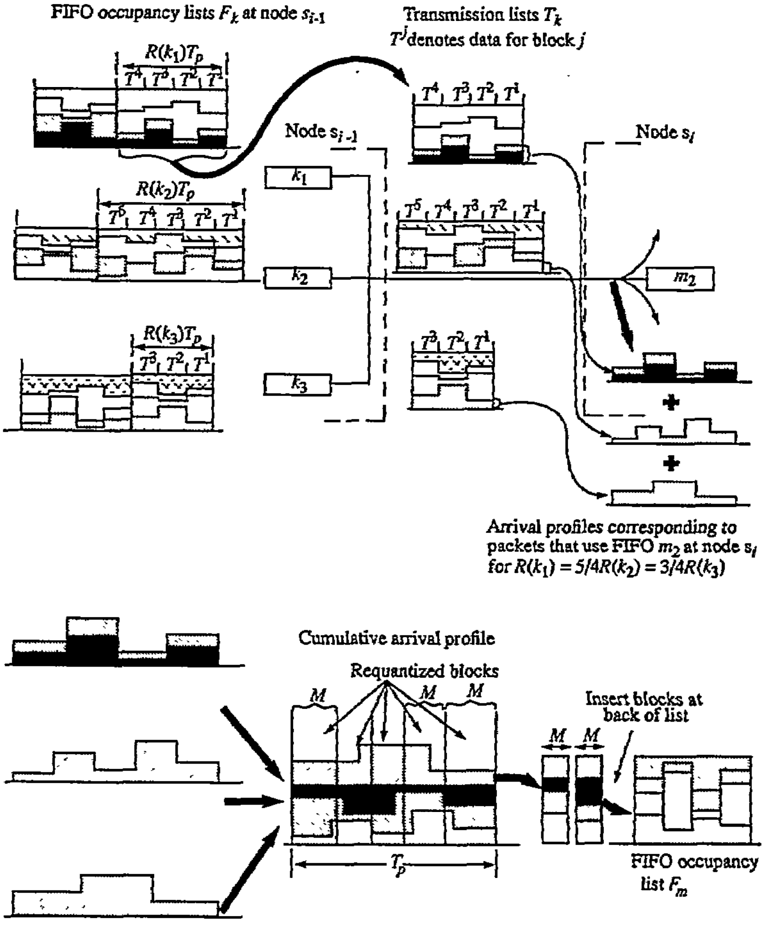
FIFO occupancy lists F_k at node s_{i-1}
Transmission lists T_k
T^j denotes data for block j
$R(k_1)T_p$
T^4 T^3 T^2 T^1
$R(k_2)T_p$
T^5 T^4 T^3 T^2 T^1
$R(k_3)T_p$
T^3 T^2 T^1
Node s_{i-1}
Node s_i
k_1
k_2
k_3
m_2
+
+
Arrival profiles corresponding to packets that use FIFO m_2 at node s_i for $R(k_1) = 5/4R(k_2) = 3/4R(k_3)$
Cumulative arrival profile
Requantized blocks
M
M
M
Insert blocks at back of list
M M
T_p
FIFO occupancy list F_m

Fig. 10. The portion of the FIFO occupancy list $\mathcal{F}_k$ that is pruned at node s_{i-1} is sent to the downstream node s_i as a *transmission list*. At the receiving node, the processor associated with FIFO m isolates that part of the transmission list that corresponds to this FIFO. The *arrival profiles* thus obtained are summed to give the *cumulative arrival profile*, which is then requantized into blocks of size M.

vertical axis representing the rates at which the arrivals from that FIFO occurred. To obtain the new elements to be added to the occupancy list of FIFO m_2, node s_i first sums the three arrival profiles, and then requantizes the resulting profile into uniform blocks of size M, which are inserted at the end of the occupancy list $\mathcal{F}_{m_2}$ of FIFO m_2. This process is illustrated graphically in Fig. 10. For the details, we refer the reader to [22].

V. Concluding Remarks

We presented the main features of the RGVC connection control protocol, assuming FIFO buffers at the switches. We introduced the concepts of *freezing* of capacity and *coupling* of capacity with buffer space, which guarantee lossless operation. We argued that the RGVC protocol leads to substantially lower connection setup times and more efficient link utilization than wait-for-reservation protocols. Although FIFO buffers do not afford the flexibility of RAM buffers, the very high speeds in multigigabit networks make them necessary because the overhead of managing RAM buffers cannot be sustained. The RGVC protocol is consistent with FIFO buffers, and has smaller control overhead than credit-based schemes. We provided measurement-based and analytic methods that use efficient list structures to perform the necessary bookkeeping, when multiple sessions share a common FIFO. The RGVC protocol is being implemented in the Thunder-and-Lightning network being built at UCSB. This implementation will enable us to test the operation of the protocol, study its performance, and validate and refine its features.

Appendix

The successive approximation algorithm for obtaining the rates $R(k)$, $k \in \{k_1, k_2, k_3\}$ for the measurement-based scheme or for the estimation-based scheme is as follows.

Initialization: Set $R(k) = C$, $k \in \{k_1, k_2, k_3\}$.

Step 1: Calculate the fractions $a(k, m)$ by averaging over $R(k)T_p$ packets, or, if less than $R(k)T_p$ packets are present, by averaging over the total number of packets present in the buffer (see Fig. 10). This is done by setting

$$q = \min\left(L, \left\lceil \frac{R(k)T_p - M_1}{M} \right\rceil + 1\right) \qquad \text{(A.1)}$$

where L is the number of elements in the FIFO occupancy list just before problem P is solved, and evaluating $a(k, m)$, $m \in \{m_1, m_2, m_3\}$ as in (A.2), shown at the bottom of the page, for the measurement-based scheme, or as in (A.3), shown at the bottom of the page, for the estimation-based scheme. In writing (A.2) and (A.3), we have used the convention that a summation with the upper limit less than the lower limit is an empty sum.

Step 2: Solve Problem P using the fractions $a(k, m)$ calculated in Step 1 to obtain $R(k)$, $k \in \{k_1, k_2, k_3\}$.

Step 3: If the new $a(k, m)$s differ from the previous ones by more than a given tolerance, go to Step 1 and iterate till convergence is reached.

References

[1] "The ATM forum traffic management specification," *ATM Forum Traffic Management AF-TM-0056.000*, Apr. 1996. (Available as ftp://ftp.atmforum.com/pub/approved-specs/aftm-0056.000.ps.)

[2] K. W. Fendick, "Evolution of controls for the available bit rate service," *IEEE Commun. Mag.*, vol. 34, pp. 35–39, Nov. 1996.

[3] F. Bonomi, D. Mitra, and J. B. Seery, "Adaptive algorithms for feedback-based flow control in high-speed, wide-area ATM networks," *IEEE J. Select. Areas Commun.*, vol. 13, pp. 1267–1283, Sept. 1995.

[4] S. E. Butner and R. Chivukula, "On the limits of electronic ATM switching," *IEEE Network*, vol. 10, pp. 26–31, Nov./Dec. 1996.

[5] S. Butner and D. Skirmont, "Architecture and design of a 40 Gigabit per second ATM switch," in *Proc. Int. Conf. Computer Design*, Austin, TX, Oct. 2–4, 1995, pp. 352–357.

[6] S. Butner, "Control structure of a 4 × 4 40 Gbit/s ATM switch," in *Proc. Int. Phoenix Conf. Computers Commun. (IPCCC'96)*, Scottsdale, AZ, Mar. 27–29, 1996, pp. 201–205.

$$a(k, m) = \begin{cases} \dfrac{1}{R(k)T_p}\left[\sum_{j=1}^{q-1} N_k^j(m) + \dfrac{N_k^q(m)}{M_q}\left(R\{k\}T_p - \sum_{j=1}^{q-1} M_j\right)\right], & \text{if } \sum_{j=1}^{q} M_j \geq R(k)T_p; \\ \dfrac{\sum_{j=1}^{q} N_k^j(m)}{\sum_{j=1}^{q} M_j}, & \text{if } \sum_{j=1}^{q} M_j < R(k)T_p \end{cases} \qquad \text{(A.2)}$$

$$a(k, m) = \begin{cases} \dfrac{1}{R(k)T_p}\left[\dfrac{\sum_{\mathcal{S} \in \mathcal{Q}(m)} N_k^q(\mathcal{S})}{M_q}\left(R\{k\}T_p - \sum_{j=1}^{q-1} M_j\right) + \sum_{j=1}^{q-1} \sum_{\mathcal{S} \in \mathcal{Q}(m)} N_k^j(\mathcal{S})\right], & \text{if } \sum_{j=1}^{q} M_j \geq R(k)T_p; \\ \dfrac{1}{\sum_{j=1}^{q} M_j}\left[\sum_{j=1}^{q} \sum_{\mathcal{S} \in \mathcal{Q}(m)} N_k^j(\mathcal{S})\right], & \text{if } \sum_{j=1}^{q} M_j < R(k)T_p \end{cases} \qquad \text{(A.3)}$$

[7] I. Cidon, I. S. Gopal, and A. Segall, "Connection establishment in high-speed networks," *IEEE/ACM Trans. Networking*, vol. 1, pp. 469–481, Aug. 1993.

[8] R. Jain, S. Kalayanraman, R. Goyal, S. Fahmy, and R. Viswanathan, "The ERICA switch algorithm for ABR traffic management in ATM networks, Part I: Description," submitted to *IEEE/ACM Trans. Networking*, Jan. 1997. (Available from http://www.cis.ohio-state.edu/jain/papers.html.)

[9] R. Jain, S. Kalyanraman, and R. Vishwanathan, "The OSU scheme for congestion avoidance using explicit rate indication," AF-TFM94-0833, Sept. 1994.

[10] M. C. St. Johns and D. Fisher, "Survey of US gigabit-class network research," in *Proc. INET'94/JENC5, Ann. Conf. Internet Soc. 5th Joint European Networking Conf.*, vol. 2, pp. 631–634.

[11] H. T. Kung, T. Blackwell, and A. Chapman, "Credit-based flow control for ATM networks: Credit update protocol, adaptive credit allocation, and statistical multiplexing," in *Proc. ACM SIGCOMM Symp. Commn., Arch., Protocols, Appl.*, 1994, pp. 101–114.

[12] S. Khorsandi and A. Leon-Garcia, "A minimal-buffer loss-freeflow control protocol for ATM networks," in *Proc. IFIP-IEEE Conf. Broadband Commun.'96*, Montreal, P.Q., Canada, Apr. 1996, pp. 161–172.

[13] M. Kato, Y. Oie, M. Murata, and H. Miyahara, "Performance analysis of reactive congestion control based upon queue length threshold values," *Perf. Eval.*, vol. 29, no. 2, pp. 105–125, Mar. 1997.

[14] H. T. Kung and K. Chang, "Receiver-oriented adaptive buffer allocation in credit-based flow control for ATM networks," in *Proc. Infocom'95*, pp. 239–252.

[15] H. T. Kung and R. Morris, "Credit-based flow control for ATM networks," *IEEE Network*, pp. 40–48, Mar./Apr. 1995.

[16] H. Ohsaki, M. Murata, H. Suzuki, C. Ikeda, and H. Miyahara, "Rate-based congestion control for ATM networks," *Computer Commun. Review*, vol. 25, no. 2, pp. 60–72, Apr. 1995.

[17] A. Peterson, T. Reynolds, and R. Nagarajan *et al.*, "3 MHz–30 GHz traveling-wave optical front-end receiver," in *Proc. OFC'95*, San Diego, CA, Feb. 1995, pp. 157–158.

[18] K. K. Ramakrishnan and P. Newman, "Integration of rate and credit schemes for ATM flow control," *IEEE Network*, pp. 49–56, Mar./Apr. 1995.

[19] K. K. Ramakrishnan and R. Jain, "A binary feedback scheme for congestion avoidance in computer networks," *ACM Trans. Computer Syst.*, vol. 8, no. 2, pp. 158–181, May 1990.

[20] M. D. Schroeder, A. D. Birell, M. Burrows *et al.*, "Autonet: A high-speed, self-configuring, local area network using point-to-point links," *IEEE J. Select. Areas Commun.*, vol. 9, pp. 1318–1335, Oct. 1991.

[21] K.-Y. Siu and H.-Y. Tzeng, "Intelligent congestion control for ABR service in ATM networks," *Computer Commun. Rev.*, vol. 24, no. 5, pp. 81–106, Oct. 1994.

[22] V. Sharma, "Efficient communication protocols and performance analysis for gigabit networks," Ph.D. dissertation, Dept. Elec. Comput. Eng., Univ. Calif., Santa Barbara, June 1997.

[23] H. Y. Tzeng and K. Y. Siu, "Comparison of performance among existing rate control schemes," *ATM Forum Contribution, 94-1078*, Nov. 1994.

[24] E. A. Varvarigos and V. Sharma, "The ERVC protocol for the Thunder and Lightning network," Tech. Rep. CIPR 95-05, Dept. Elec. Comput. Eng., Univ. California, Santa Barbara, Jan. 1995. (Available from spectses.ece.ucsb.edu:/pub/manos/protocols. Filename: ervc_cipr9505.ps.)

[25] ——, "Lossfree communication in high speed networks," in *Proc. IEEE Sing. Int. Conf. Networks, SICON'95*, Singapore, July 3–7, 1995, pp. 230–236.

[26] Y. T. Wang and B. Sengupta, "Performance analysis of a feedback congestion control policy under nonnegligible propagation delay," *Comp. Commun. Rev.*, vol. 21, no. 4, pp. 149–157, Sept. 1991.

Emmanouel A. Varvarigos (M'92) received the Diploma in electrical engineering from the National Technical University of Athens, Greece, in 1988 and the M.S. degree, electrical engineer, and Ph.D. degrees in electrical engineering and computer science from the Massachusetts Institute of Technology, Cambridge, in 1990, 1991, and 1992, respectively.

In 1990, he conducted research on optical fiber communications at Bell Communications Research, Morristown, NJ. He is currently an Assistant Professor at the Department of Electrical and Computer Engineering, University of California, Santa Barbara. His research interests are in the areas of high-speed data networks, parallel and distributed computation, and mobile communications.

Dr. Varvarigos received the first Panhellenic prize in the Greek Mathematic Olympiad in 1982. He is a member of the Technical Chamber of Greece.

Vishal Sharma (S'93) received the B.Tech. degree in electrical engineering from the Indian Institute of Technology, Kanpur, in 1991, the M.S. degrees in computer engineering and in signals and systems in 1993, and the Ph.D. degree in electrical and computer engineering from the University of California, Santa Barbara, in 1997.

In the summer of 1992, he worked at the Digital Technology Research Laboratory in Motorola's Corporate Research and Development Center, Schaumburg, IL, where he investigated algorithms for real-time resizing of decompressed video sequences. His current research interests are in protocols, architectures, and performance analysis for terrestrial or satellite-based broadband networks, and all-optical networks. Since September 1997, he has been with Qualcomm, Inc., San Diego, CA, where he is associated with the Globalstar project and with other data-transmission-over-wireless initiatives.

Dr. Sharma is active in the IEEE Computer and Professional Communication Societies, and is a Student Member of the ACM SIGCOMM and SIGDOC.