ATM ABR Multipoint-to-point Connections and Fairness Issues*

Sonia Fahmy, Raj Jain, Rohit Goyal, and Bobby Vandalore
The Ohio State University,

Raj Jain is now at Washington University in Saint Louis, jain@cse.wustl.edu http://www.cse.wustl.edu/~jain/

Abstract

In multipoint-to-point connections, multiple sources can concurrently send data to the same destination. A crucial concern in this case is how to define fairness within a multicast group, and among multicast groups and unicast connections. The multipoint connection can have the same identifier on each link, and senders might not be distinguishable. Many Available Bit Rate (ABR) rate allocation algorithms implicitly assume that there is only one sender in each connection, which does not hold for multipoint-to-point cases. We give four fairness definitions for multipoint connections based on connections, senders or flows, and we discuss the tradeoffs involved. In addition, we show that ATM bandwidth allocation algorithms need to be adapted to give fair allocations for multipoint-to-point connections.

Keywords: ATM networks, fairness, traffic management, congestion control, available bit rate (ABR) service, multipoint communication, multipoint-to-point connections, multicasting

1 Introduction

Multipoint communication is the exchange of information among multiple senders and multiple receivers. Multipoint support in Asynchronous Transfer Mode (ATM) networks is essential for efficient duplication, synchronization and coherency of data (refer to figure 1). Examples of multipoint applications include audio and video conferencing, server and replicated database synchronization, advertising, and data distribution applications. Multipoint-to-point connections are especially important for overlaying IP networks and simplifying end systems and edge devices [28]. In this case, only one multipoint-to-point connection needs to be set up even if there are multiple senders. This reduces the connection management and control overhead over using multiple point-to-point connections.

*An earlier version of this paper was presented at the Performance and Control of Network Systems II Conference, Wai Sum Lai, Robert B. Cooper, Editors, Proceedings of SPIE (The International Society for Optical Engineering) Vol. 3530, Boston, Massachusetts, 2-4 November 1998.

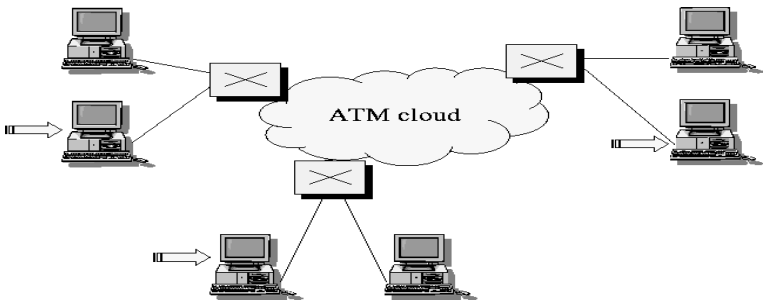

Figure 1: Multipoint communication in ATM

An efficient and flexible ATM multipoint service is key to the success of ATM networks. Several issues need to be addressed in the ATM multipoint service definition, such as routing, connection management, and traffic management. In this paper, we focus on the traffic management issues in the case of *multiple senders*. Specifically, we tackle the definition of fairness, and the available bit rate (ABR) flow control problem for multipoint-to-point connections.

ATM networks currently offer five service categories: constant bit rate (CBR), real-time variable bit rate (rt-VBR), non-real time variable bit rate (nrt-VBR), available bit rate (ABR), and unspecified bit rate (UBR).[1] Switches generally service CBR and VBR traffic in preference to ABR traffic. The capacity left-over is fairly divided among the active ABR sources [10]. The key attractive features of ABR are that (1) it gives sources low cell loss guarantees, (2) minimizes queuing delay, (3) provides possibly non-zero minimum rate guarantees, (4) utilizes bandwidth and buffers efficiently, and (5) gives the contending sources *fair* shares of the available resources. (Thus it is ideal for data distribution applications and also performs well for real-time applications with the appropriate implementation and parameter choices. It is superior to CBR and VBR because it can utilize available bandwidth and buffers more efficiently, and avoids congestion through feedback.) The most commonly adopted fairness definition is the max-min fairness definition [13]. Intuitively, this means that all sources bottlenecked at the same node are allocated equal rates. This definition was developed for point-to-point connections, and in this paper, we explore multipoint extensions.

For *point-to-multipoint* ABR connections, the source is usually controlled to the minimum rate supported by all the leaves of the multicast tree, as the leaves usually cannot tolerate losses [8, 22, 24, 27]. Therefore, the extension of the fairness definition to point-to-multipoint connections is straightforward. With multipoint-to-point and multipoint-to-multipoint connections, however,

[1] In addition to these, the guaranteed frame rate (GFR) service is being finalized.

the implicit assumption that each connection has only one source is no longer valid. The fairness definition relies on the application type and the pricing methods.

In this paper, we define four methods for computing the fair allocations for multipoint-to-point virtual connections (VCs), and discuss the necessary modifications to switch schemes to give these allocations. The remainder of this paper is organized as follows. Section 2 provides a brief overview of ABR flow control. Section 3 discusses ATM multipoint support and the solutions to the cell interleaving problem. Section 4 summarizes related work on fairness definitions and previous multipoint-to-point ABR algorithms. We present our fairness definitions in section 5, and show their operation, merits and drawbacks with the aid of examples. We then discuss several design issues and examine how switch schemes need to be adapted to give fair allocations in section 6. The paper concludes with a summary of the issues involved.

2 ABR Flow Control

The ABR service frequently indicates to the sources the rate at which they should be transmitting. The feedback from the switches to the sources is indicated in resource management (RM) cells which are generated periodically by the sources and turned around by the destinations. Figure 2 illustrates this operation.

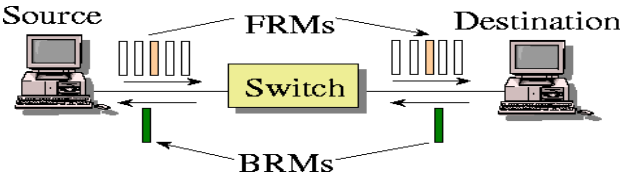

Figure 2: Resource management cells in an ATM network

The RM cells contain the source current cell rate (CCR), in addition to several fields that can be used by the switches to provide feedback to the sources. Among these fields, the explicit rate (ER) field indicates the rate that the network can support for this connection at that particular instant. At the source, the ER field is initialized to a rate no greater than the PCR (peak cell rate). Each switch on the path from the source to the destination reduces the ER field to the maximum rate it can support [14].

A component c_j is said to be *downstream* of another component c_i in a certain connection if c_j

is on the path from c_i to the destination. In this case, c_i is said to be *upstream* of c_j. The RM cells flowing from the source to the destination are called forward RM cells (FRMs) while those returning from the destination to the source are called backward RM cells (BRMs). When a source receives a BRM cell, it computes its allowed cell rate (ACR) using its current ACR value, and the ER field of the RM cell [15].

3 ATM Multipoint Support and Cell Interleaving Solutions

ATM multipoint communication is the focus of extensive research at the ATM Forum and at the International Telecommunications Union (ITU). The Internet Engineering Task Force (IETF) has also studied the mapping of IP multicast to ATM networks. ATM user to network interface (UNI) signaling currently supports multicast via *point-to-multipoint VCs*. The ATM UNI 3.1 signaling standard supports the source-based tree approach for multicast data distribution, and uses root-initiated joins for multicast tree construction. Thus, only the root can setup the point-to-multipoint connection, add leaves, and send ATM cells. Receiver or leaf initiated join (LIJ) avoids the bottleneck at the root, so UNI 4.0 signaling supports such joins. Pure multipoint-to-point and multipoint-to-multipoint services are not yet supported, although the signaling and PNNI working groups at the ATM Forum have defined the operation of multipoint-to-point connections for the UBR service category. A number of proposals for supporting ATM multipoint connections as a single shared tree can be found in [11, 12].

In ATM networks, the virtual path identifier (VPI)/virtual connection identifier (VCI) fields in the cell header are used to switch ATM cells. The ATM adaptation layer (AAL) at the sender segments packets into ATM cells, marking the last cell of each packet. The AAL at the receiver uses the VPI/VCI fields and the end of packet marker to reassemble the data from the cells received.

ATM adaptation layer 5 (AAL5), which is used for most data traffic, does not introduce any multiplexing identifier or sequence number in ATM cells. If cells from different senders are merged and interleaved on the links of a multipoint connection (implemented as a shared tree), the AAL5 at the receiver cannot assemble the data. This is because all traffic within the group uses the same VPI/VCI. The AAL5 layer uses the end-of-message bit to determine the end of each packet, but since the cells of different packets are interleaved, all the packets may get corrupted, as illustrated

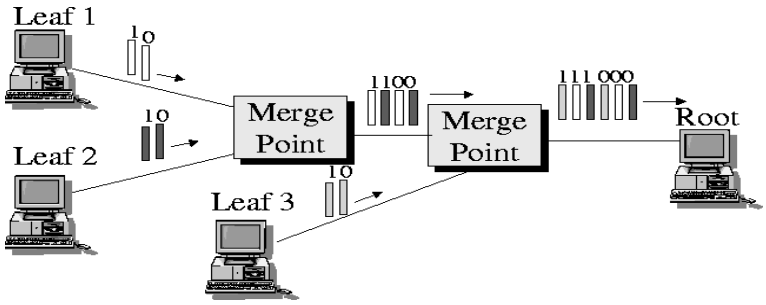

Figure 3: The cell interleaving problem

in figure 3. The identity of the sender is not indicated in each cell. Hence, alternate solutions must be implemented.

Solutions to this problem either (a) entirely avoid merging, or (b) prevent interleaving of cells of packets originating from different sources on the same multipoint connection after merging, or (c) provide enough information in the cell headers to enable the receivers to reassemble the packets even if their cells are interleaved.

The solutions proposed to the cell interleaving problem include:

1. **AAL3/4:** AAL3/4 can be used instead of AAL5. AAL3/4 contains a 10-bit multiplexing identifier (MID) field, part of which can be used to distinguish the senders in the multipoint VC. This can make switching fast and connection management simple. However, AAL3/4 suffers from excessive overhead and is not well supported. An alternative AAL, AAL5+, was proposed in [18].

2. **VC mesh:** Another solution is to overlay point-to-multipoint VCs to create multipoint-to-multipoint, forming a VC mesh [1]. In this case, cells from different senders can be differentiated based on their VPI/VCI fields. This solution does not scale and requires N one-to-many VCs for N senders.

3. **Multicast servers (MCSs):** In this case, all senders send to the MCS, which forwards data on a point-to-multipoint VC [29]. This approach is simple. The problem with it is that it is inefficient, and the MCS needs large amounts of buffering. In addition, the MCS can become a single point of congestion, which makes it difficult to guarantee quality of service requirements.

4. **Election:** Election can be used to coordinate senders. In this approach, a sender must acquire a control message before it can transmit data, and there is only one token in each VC. Hence, only one sender can transmit at a time, and no cell interleaving can possibly occur. This approach is used in the SMART scheme [11]. Although this mechanism is feasible, the overhead and delay of the scheme are high.

5. **VC merge:** The VC merge approach buffers cells of other packets at the switch until all cells of the current packet go through (as shown in figure 4). The technique is also called "cut-through forwarding," and it is used in the SEAM [12] and ARIS schemes. It entails the implementation of a packet-based scheduling algorithm at the merge point, and maintaining separate queues for each sender. The AAL5 end-of-message bit is used to signal to the switch that a packet from a different port can now be forwarded. The approach is fast and simple, but requires more memory at the switches, and adds to the burstiness and latency of traffic. An analysis in [35] shows that the effect of VC merge on traffic is minimal.

6. **VP merge:** This approach uses multipoint virtual paths (VPs). Only the VPI field is used for switching cells of a multipoint connection, and the VCI field is used to uniquely identify the sender. Connection management is simple in this case, but the approach requires receivers to have global assignment of VCs within VPs. The main problem, however, is that VPs should not be used by end-systems, since network providers use VPs for aggregation in the backbone. Finally, there are only $2^{12} = 4096$ unique VPI values possible at each hop, and hence it is possible to run out of VPI values.

7. **Variable VP merge:** Different VPI field sizes are used in this approach [32]. The switches support both 12-bit VPI fields, as well as 18-bit VPI fields. Distributed schemes to assign globally unique VCIs within each VP are proposed using collision avoidance. This approach overcomes the VP scarcity problem of VP merge, but still has the problem of using VPs. Furthermore, it complicates the switch design since two distinct VP tables need to be maintained.

8. **Sub-channel multiplexing:** A sub-channel is a "channel within a VC." Each sub-channel can be assigned an identifier called the sub-channel number to distinguish between multiple

sub-channels in a VC [30]. Four bits from the Generic Flow Control (GFC) bits in the ATM cell header can carry this number. Each burst of cells is preceded by a "start" resource management (RM) cell, and followed by an "end" RM cell. The sub-channel is allocated on the "start" cell and released on the "end" cell. Sub-channel identifiers can change at every switch. This approach allows dynamic sharing by using on-the-fly mapping of packets to sub-channels. However, four bits only allow up to fifteen concurrent senders (sub-channel number hexadecimal FF indicates an idle sub-channel). If no sub-channel is available, the burst of cells is lost, so this solution may not be scalable.

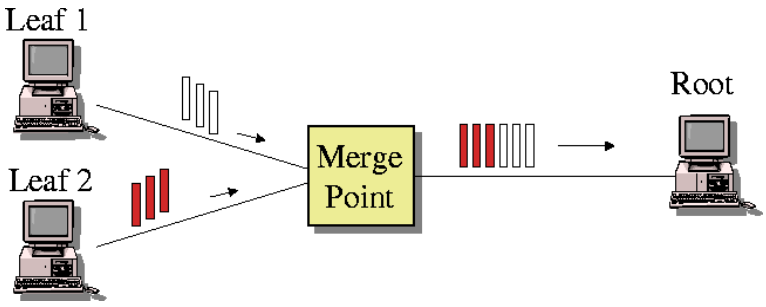

Figure 4: The VC merge approach

Although each of the approaches has its own merits and drawbacks, VC merge and VP merge are the most popular approaches today. This paper emphasizes the issues involved if VC merge and VP merge are implemented.

4 Fairness and Existing Multipoint Algorithms

The optimal operation of a distributed shared resource is usually given by a criterion called the *max-min allocation* [13]. This fairness definition is the most commonly accepted one, though other definitions are also possible.

Definition: Given a configuration with n contending sources, suppose the i^{th} source is allocated a bandwidth x_i. The allocation vector:

$$\{x_1, x_2, \ldots, x_n\}$$

is *feasible* if all link load levels are less than or equal to 100%. □

Definition: Max-min allocation: Given an allocation vector $\{x_1, x_2, \ldots, x_n\}$, the source that is getting the least allocation is, in some sense, the "unhappiest source". We need to find the feasible vectors that give the maximum allocation to this unhappiest source. Now we remove this "unhappiest source" and reduce the problem to that of the remaining $n-1$ sources operating on a network with reduced link capacities. Again, we find the unhappiest source among these $n-1$ sources, give that source the maximum allocation, and reduce the problem by one source. We repeat this process until all sources have been allocated the maximum that they can get. □

Intuitively, this means that all sources bottlenecked on the same link get equal rates, and if a source cannot utilize its fair share, the left over capacity is shared fairly among those who can use it. An extension of this definition guarantees a minimum cell rate (MCR), and shares the left-over capacity in a weighted manner. This is called the general weighted fair allocation [31].

Definition: General weighted fair allocation: Given a weight vector $\{w_1, w_2, \ldots, w_n\}$ that denotes the weight to be given to each source, and an MCR vector $\{MCR_1, MCR_2, \ldots, MCR_n\}$ denoting the minimum cell rates for each source, the allocation for each source is denoted by:

$$x_i = MCR_i + \frac{w_i \times (\text{Capacity} - \sum_{i=1}^{n} MCR_i)}{\sum_{j=1}^{n} w_j}$$

□

4.1 Multiple Receivers

In the case of multiple receivers, the rate at which the source can send depends on the rates that can be supported by the all links on the path from the source to all the receivers. The choice of fairness definition to adopt is application specific [9]. The most commonly adopted fairness definition for the ABR service is a simple extension of the max-min fairness definition such that the source sends at the bottleneck rate indicated by switches on all the paths. In other words, the source attempts to minimize loss on all the branches by sending at no more than the rate that can be supported by the slowest branch. This is the best approach if none of the receivers can tolerate loss.

Since this approach can result in low link utilization on all the multicast tree branches other than the most congested branch, several alternative approaches have been proposed. These approaches attempt to maximize the *intra-session* fairness, or providing a receiver with a rate according to its

capabilities and the capacity of the path leading to it, regardless of the capabilities of the other receivers.

A popular approach to maximize this intra-group fairness is using multiple multicast groups. The source transmits the data to multiple groups at different rates. Each receiver subscribes to the group where the data is being sent at or lower than the bottleneck rate on the path to that receiver. Receivers can dynamically unsubscribe and subscribe to groups when conditions change. Several algorithms have been based on this idea [2, 4]. Layering the transmitted data [21, 33], using forward error correction [33], and using hierarchical approaches [5, 9, 20] have also been proposed in the context of reliable multicast transport and congestion control.

A number of studies have also proposed that the sender transmit at a rate higher than that requested by the slowest branch, in order to maximize a certain fairness metric. This "quasi-reliability" can be used for information dissemination applications where the receiver can "tune" the level of reliability requested [36]. Another metric that can be maximized is the *inter-receiver* fairness, which is defined as the ratio between the minimum of the sender rate and the bottleneck rate supported for a branch, to the maximum of the two. This can be maximized in a weighted manner for all branches, as long as the application-specific loss tolerance is met [17].

Inter-session fairness (or inter-group) fairness is defined as providing a receiver with a rate according to its capabilities and the capacity of the path leading to it, regardless of the capabilities of the other receivers in the group and other groups [19]. A similar concept called bounded fairness or *essential fairness* is proposed among TCP and point-to-multipoint connections in [34]. If the throughput of TCP is denoted by λ_{TCP} and that of multicast connections is denoted by λ_m, then a multicast session is essentially fair if:

$$a \times \lambda_{TCP} < \lambda_m < b \times \lambda_{TCP}$$

4.2 Multiple Senders

Little work has been done to define fairness and traffic management rules for multipoint-to-point connections. Multipoint-to-point connections require feedback to be returned to the appropriate sources at the appropriate times. As illustrated by figure 5, the bandwidth requirements for a VC after a merge point is the sum of the bandwidths used by all senders whose traffic is merged. This

is because the aggregate data rate after a merge point is the sum of all incoming data rates to the merge point [16]. Similarly, the number of RM cells after merging is the sum of those from different branches. Hence, if all sources use the same RM cells to data cells ratio (as denoted by the parameter Nrm), the ratio remains the same after merging.

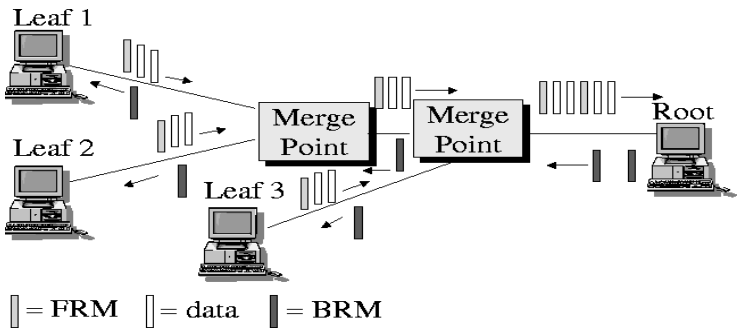

Figure 5: Multipoint-to-point connections

Ren and Siu [25] describe an algorithm for multipoint-to-point congestion control that allocates bandwidth fairly among contending sources. The algorithm assumes that a multipoint-to-point VC is defined as a shared tree, and that VC merging is employed to prevent the cell interleaving problem. The authors have proved that given a max-min fair point-to-point rate allocation algorithm, their proposed multipoint-to-point extension framework is also max-min fair among sources. The authors do not discuss why fairness among sources is the best approach.

The idea of Ren and Siu's algorithm is similar to point-to-multipoint algorithms [8, 22, 24, 27]. The algorithm operates as follows. When a forward resource management (FRM) cell originating at a leaf is received at the merge point, it is forwarded to the root, and the merge point returns a backward resource management (BRM) cell to the flow which had sent the FRM cell. The explicit rate in the BRM cell is set to the value of a register called MER (explicit rate), maintained at the merge point for each VC. The MER register is then reset to the peak cell rate. When a BRM cell is received at the merge point, the ER value in the BRM is used to set the MER register, and the BRM cell is discarded. This algorithm, however, can cause rate oscillations, and is complex to implement due to the RM cell turn-around at the merge points.

The same authors remedy these problems in [26]. The algorithm maintains a bit at the merge point for each of the flows being merged. The bit indicates that an FRM has been received from this flow after a BRM had been sent to it. Therefore, when an FRM is received at the merge point, it

is forwarded to the root and the bit is set, but the RM cell is not turned around as in the previous algorithm. When a BRM is received at the merge point, it is duplicated and sent to all the branches that have their bit set, and then the bits are reset. This saves the overhead that the merge point incurs when it turns around RM cells, since only destinations turn around RM cells in this case. The problem with Ren and Siu's work is that it does not clearly state which types of rate allocation algorithms the proposed multipoint extension works for. In fact, the extension does not work for many popular ABR schemes that perform per VC accounting, since this is no longer equivalent to per-source accounting, as discussed in section 6.

Recently more complex algorithms have been developed [3, 23] for multipoint-to-point and multipoint-to-multipoint connections respectively. In [3], the algorithm proposed attempts fairness among the sources as in [25]. The algorithm in [23] adds a *weight* in RM cells to allow scaling of the rates to give the appropriate allocations to sources. The throughput of a unicast source is given a predetermined weight with respect to that of a sender in a multicast session. This technique adds more flexibility at the expense of complexity in RM cells and processing.

5 Fairness Extension for Multipoint-to-Point Connections

In this section, we define fairness for the multiple sender case, and show examples of the operation of our definitions. In addition, we discuss the merits and drawbacks of each fairness definition.

5.1 Fairness Definitions

Sources, VCs and flows. Before giving the fairness definitions, we need to distinguish among sources, VCs and flows. Figure 6 shows a configuration with 2 VCs. One of the VCs is a point-to-point VC, while the other is a multipoint-to-point VC. The senders in the multipoint-to-point VC are indicated by dark-colored filled circles, while the sender in the point-to-point VC is denoted by the unfilled circle. At the second switch, traffic from 4 sources, but only 2 VCs, is being switched to the output port. Note, however, that the second switch can distinguish 3 input flows (the point-to-point sender and 2 flows of the multipoint-to-point connection). The 2 sources whose traffic was merged at the first switch constitute a single flow at the second switch, since they cannot be distinguished downstream of their merge point. Two of the input flows that can be distinguished

at the second switch belong to the same VC, while the third flow belongs to a different VC. The second switch merges the two flows of the same VC, and they become a single flow downstream of the second switch.

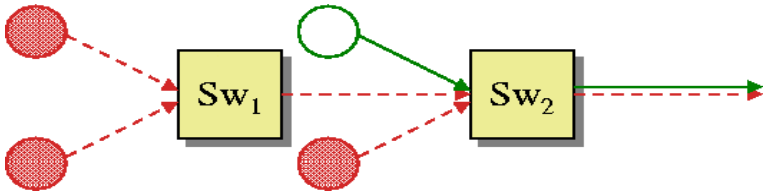

Figure 6: Source versus connection versus flow

Throughout the rest of the paper, we use max-min fairness as the underlying fairness definition. However, the definitions we give apply for any underlying definition, e.g., general weighted fairness with minimum rate guarantees. Assume that a configuration is a set of sources, destinations, and switches, interconnected with links of given distances and bandwidths. We use the following notation:

n	denotes the number of sources in a given configuration
m	denotes the number of connections in a given configuration
f	denotes the number of flows being switched to an output port of a switch
x_i	denotes the allocation given to the i^{th} source
y_j	denotes the allocation given to the j^{th} connection
z_k	denotes the allocation given to the k^{th} flow
N	denotes the number of senders in a certain multipoint-to-point connection
F	denotes the number of flows in a certain multipoint-to-point connection

Source-based. Source-based fairness gives fairness among all sources, regardless of which virtual VC each source belongs to. Each N-to-one connection is treated the same as N one-to-one connections.

Definition: *Source-based* fairness divides bandwidth fairly among active sources as if they were sources in point-to-point connections, ignoring group memberships.

The allocation vector $\{x_1, x_2, \ldots, x_n\}$ is determined based on applying the underlying fairness definition for all active sources x_i. □

With source-based fairness, VCs that have a larger number of concurrently active senders get more bandwidth than VCs with less concurrent senders on the same link. The resource allocation can be unfair *among the VCs*.

VC/source-based. Bandwidth allocation can be performed for all VCs, and allocations to the sources in the same VC can be fair within the VC.

Definition: *VC/source-based* fairness first gives fair bandwidth allocations among the VCs, and then fairly allocates the bandwidth of each VC among its sources.

The allocation vector $\{y_1, y_2, \ldots, y_m\}$ is determined based on applying the underlying fairness definition for all active VCs y_j.

The allocation vector $\{x_1, x_2, \ldots, x_N\}$ for each of the VCs is determined based on the applying the underlying fairness definition for all active sources in the VC, and based on the capacity available for the VC. □

Flow-based. A third possibility is *flow-based* fairness. Intuitively, each VC coming on an input port (link) is considered a separate flow. Hence, two VCs coming on the same input port are considered two separate flows, and traffic coming from two *different* input ports on the same VC (and being merged at the switch) is also considered as two separate flows. *The key advantage of this technique is that a switch can easily distinguish the flows.*

Definition: *Flow-based* fairness gives fair allocations for each active flow, where a flow is a VC coming on an input link. Formally, we define the number of *flows* for an output port as the sum of the number of active VCs sending to this output port, for each of the input ports of the switch:

$NumFlows_j$, $j \in OutputPorts$ =

$\forall i, i \in InputPorts, \sum_i$ Number of active VCs coming on port i and being switched to port j

The allocation vector $\{z_1, z_2, \ldots, z_f\}$ is determined based on the applying the underlying fairness definition for all active flows z_k being switched to an output port. □

For example, assume that at a certain switch, traffic is coming from three different input ports (ports 1, 2, and 3). The traffic is being switched to the same output port (port 4). One of the input ports (port 2) has two VCs sending to port 4, while each of the other two ports (ports 1 and 3) has only one VC sending to port 4 (may be the same VC, but different senders). Then the number of flows at port 4 would be considered as 2 (port 2), plus 1 (port 1), plus 1 (port 3), equals four. Note that the flow-based allocation is "local" since there is no global notion of flow (two flows can merge and become a single flow). We will later see that the flow-based fairness definition suffers from a major drawback.

VC/flow-based. The flow-based definition can be also be adopted within each VC in the VC-based approach, which we call VC/flow-based fairness.

Definition: *VC/flow-based* fairness first gives fair bandwidth allocations among the VCs, and then fairly allocates the bandwidth of each VC among its flows.

The allocation vector $\{y_1, y_2, \ldots, y_m\}$ is determined based on applying the underlying fairness definition for all active VCs y_j.

The allocation vector $\{z_1, z_2, \ldots, z_F\}$ for each of the VCs is determined based on the applying the underlying fairness definition for all active flows z_k in the VC, and based on the capacity available for the VC. □

The examples presented next will clarify the differences between the various fairness definitions.

5.2 Examples

We explain multipoint fairness with the aid of two examples. We use *max-min fairness* as the underlying fairness definition throughout. We postpone the discussion of how to design and implement them to section 6. The first example illustrates a downstream bottleneck situation, while the second one shows an upstream bottleneck, to illustrate the allocation of capacity left-over by connections bottlenecked elsewhere.

5.2.1 Example 1

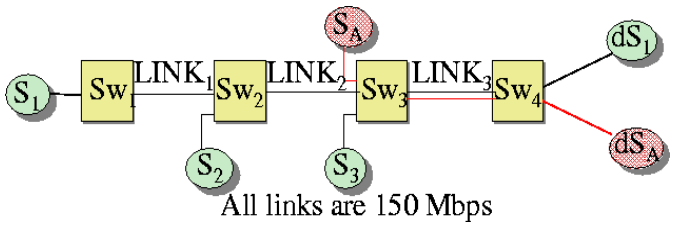

Figure 7: Example multipoint-to-point configuration with a downstream bottleneck

Figure 7 illustrates a configuration with two VCs: one of the VCs is a multipoint-to-point VC with three senders and one receiver, and the other is a point-to-point VC. Sources S_1, S_2, and S_3 are sending to destination dS_1, and source S_A is sending to destination dS_A. All links are approximately 150 Mbps (after SONET overhead is accounted for). Clearly, all four sources are sharing a bottleneck link ($LINK_3$) between $Switch_3$ and $Switch_4$. **The aim of this example**

is to show the division of the 150 Mbps capacity of this bottleneck link among the sources.

Source-based Definition. In this case, we disregard which sources belong to which connections, and simply treat this as a regular "four sources on a single bottleneck" situation. Applying the max-min fairness definition among sources, the allocations computed are:

$\{S_1, S_2, S_3, S_A\} \leftarrow \{37.5,\ 37.5,\ 37.5,\ 37.5\}$

Each of the four sources is allocated $\frac{1}{4} \times 150 = 37.5$.

Observe, however, that on $LINK_3$, the multipoint-to-point VC is getting 3 times as much bandwidth as the point-to-point VC. If there were 100 concurrent senders in the multipoint-to-point VC, the VC obtains 100 times as much bandwidth as the point-to-point VC. In essence, the bandwidth allocated to a multipoint-to-point VC with N concurrent senders all bottlenecked on a certain link would be N times the bandwidth for a point-to-point VC bottlenecked on that same link, and N/K times that for a K-sender multipoint-to-point VC bottlenecked on the same link.

VC-based Definition: VC/Source. If a VC-based definition is adopted, we are essentially dividing up the fair allocation computation process into two phases. In the first phase, we ignore the number of senders in each VC, and simply applying the max-min fairness computation to the VCs at each node. In the second phase, we take each multipoint-to-point VC separately and divide up its allocation max-min fairly among the senders in that VC. This process is repeated for each multipoint-to-point VC.

According to this definition, the allocation vector for the example above would be:

$\{S_1, S_2, S_3, S_A\} \leftarrow \{25,\ 25,\ 25,\ 75\}$

This is because both of the VCs are bottlenecked at $LINK_3$, so each VC is allocated half of the available bandwidth ($\frac{1}{2} \times 150 = 75$). Then, for the multipoint-to-point VC, we see that $LINK_3$ is again the bottleneck, so each of the three sources gets one third of the bandwidth allocated to this VC ($\frac{1}{3} \times 75 = 25$).

Flow-based Definition. Recall that a flow was defined as a VC coming on an input port. According to this definition, the number of flows on $LINK_3$ is three, and hence each of the flows gets one third of the bottleneck bandwidth ($\frac{1}{3} \times 150$). This bandwidth is then divided equally among the two flows detected at the output port of $Switch_2$, producing the allocation vector:

$\{S_1, S_2, S_3, S_A\}$ $\leftarrow$$\{25,\ 25,\ 50,\ 50\}$

Clearly, the allocation suffers from the "beat-down problem" commonly observed with bit marking switches. We discuss this further in section 5.3.

VC-based Definition: VC/Flow. If we use the VC-based approach, but, instead of dividing the bandwidth among the senders in the same VC max-min fairly, we divide the bandwidth max-min fairly among the flows in the VC, a different allocation vector is obtained. For the example above, the allocation vector would be:

$\{S_1, S_2, S_3, S_A\}$ $\leftarrow$$\{18.75,\ 18.75,\ 37.5,\ 75\}$

This is because the bandwidth is divided max-min fairly among the two VCs at $Switch_3$, giving 75 Mbps to each VC. For the multipoint-to-point VC, $Switch_3$ divides the 75 Mbps equally among the two flows in that VC, so the flow originating from S_3 is allocated $\frac{1}{2} \times 75 = 37.5$ Mbps. $Switch_2$ divides the 37.5 Mbps that $Switch_3$ had allocated to the flow consisting of S_1 and S_2 equally among these two sources, each obtaining $\frac{1}{2} \times 37.5 = 18.75$ Mbps.

5.2.2 Example 2

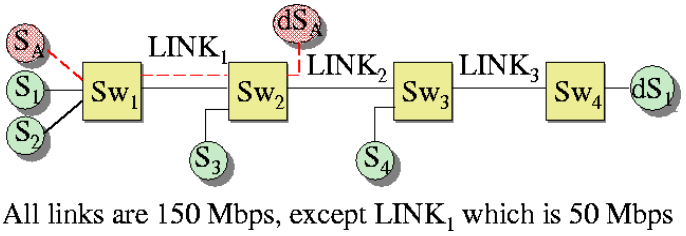

Figure 8: Example multipoint-to-point configuration with an upstream bottleneck

Figure 8 illustrates a configuration with two VCs: one of the VCs is a multipoint-to-point VC with four senders and one receiver, and the other is a point-to-point VC. Sources S_1, S_2, S_3 and S_4 are sending to destination dS_1, and source S_A is sending to destination dS_A. All links are approximately

150 Mbps (after SONET overhead is accounted for), except for the link between $Switch_1$ and $Switch_2$ ($LINK_1$) which is only 50 Mbps. Clearly, sources S_1, S_2 and S_A are bottlenecked at $LINK_1$, while sources S_3 and S_4 are bottlenecked at $LINK_3$. **The aim of this example is to illustrate the allocation of the capacity left over by sources bottlenecked on $LINK_1$ to the sources bottlenecked on $LINK_3$.**

Source-based Definition. The allocation vector according to the source based definition is:

$\{S_1, S_2, S_3, S_4, S_A\} \leftarrow \{16.67,\ 16.67,\ 58.33,\ 58.33,\ 16.67\}$

This is because each of sources S_1, S_2 and S_A is allocated one third of the bandwidth of $LINK_1$. At $LINK_3$, the $50 \times \frac{2}{3} = 33.33$ Mbps used by sources S_1 and S_2 is subtracted from the available bandwidth, and the remaining capacity (116.67 Mbps) is equally divided upon sources S_3 and S_4.

VC-based Definition: VC/Source. According to the VC/Source definition, the allocation vector for the example above would be:

$\{S_1, S_2, S_3, S_4, S_A\} \leftarrow \{12.5,\ 12.5,\ 62.5,\ 62.5,\ 25\}$

This is because each of the VCs is allocated half of the bandwidth on $LINK_1$, and this bandwidth is divided equally among S_1 and S_2 of the multipoint VC. On $LINK_3$, the remaining capacity ($150 - 25 = 125$ Mbps) is divided max-min fairly among the sources within the multipoint-to-point VC.

Flow-based Definition. Here the allocation vector is:

$\{S_1, S_2, S_3, S_4, S_A\} \leftarrow \{16.67,\ 16.67,\ 41.67,\ 75,\ 16.67\}$

This is because $Switch_3$ detects two flows on $LINK_3$, and allocates half of the capacity to each flow (hence, source S_4 is allocated half of $LINK_3$ bandwidth). $Switch_1$ divides the 50 Mbps equally among the three flows sharing $LINK_1$ (each of S_1, S_2 and S_A gets $\frac{1}{3} \times 50 = 16.67$). $Switch_2$ divides the 75 Mbps (that $Switch_3$ had allocated to the flow emerging from it) equally among the flow from S_3 and the flow from $Switch_1$, but detects that one of the flows (that from $Switch_1$, i.e., S_1 and S_2) is only using 33.33 Mbps, so it allocates the remaining $75 - 33.33 = 41.67$ Mbps to source S_3.

VC-based Definition: VC/Flow. According to the definition, the allocation vector for this case is:

$\{S_1, S_2, S_3, S_4, S_A\} \leftarrow \{12.5,\ 12.5,\ 50,\ 75,\ 25\}$

$Switch_1$ divides the available 50 Mbps equally among the two VCs (giving each 25 Mbps), and divides the bandwidth of the multipoint-to-point VC equally among the two flows in that VC (each getting 12.5 Mbps). $Switch_3$ divides the bandwidth fairly among the two flows, allocating 75 Mbps to the flow from S_4 and 75 Mbps to the flow from $Switch_2$. $Switch_2$ detects that the flow from S_1 and S_2 is only using 25 Mbps, so it allocates the remaining 50 Mbps to the other flow (source S_3).

5.3 Merits and Drawbacks of the Different Definitions

The examples above illustrate how fairness based upon the concepts of source, VC, and flow can give different allocations. Appropriate fairness definitions are affected by application types and pricing issues.

Flow and VC/flow. The flow-based method is not max-min fair if we view an N-to-one connection as N one-to-one connections, since the same flow can combine more than one source. The flow-based and VC/flow-based methods inherently suffer from the beat-down problem. The "beat-down problem" means that sources whose flow travels a larger number of hops are allocated less bandwidth than those traveling a smaller number of hops, even if both flows have the same bottleneck. In flow-based fairness for multipoint connections, sources whose flow traverses a larger number of merge points are allocated less bandwidth than those traversing a smaller number of merge points. This is because once the flows of twp sources are merged at a switch, they are considered as a single flow at all downstream switches. "Example 1" above clearly shows that sources 1 and 2 are allocated less bandwidth because their flows traverse a larger number of merge points than source 3.

One may, however, argue that it may be desirable to favor sources traversing a smaller number of merge points, since these are more likely to encounter less bottlenecks. For example, if a user in New York city is fetching some web pages from a server several hops away (say in Germany), he expects to wait longer than if he is fetching web pages within his same local network. Thus, although flow-based fairness is unfair to sources whose traffic is merged multiple times, it may be acceptable in many practical situations. The VC/flow-based fairness is max-min fair with respect

to VCs, but within the same VC, it favors sources whose traffic goes through a smaller number of merge points.

Source and VC/source. Source-based fairness entirely ignores membership of sources to connections, and divides the available bandwidth fairly among the currently active sources. If pricing is based upon sources, this mechanism is good, since allocation is fair among sources.

If pricing is based on connections, however, a VC with a hundred concurrent senders should not be allocated hundred times the bandwidth of a point-to-point connection bottlenecked on the same link. Source-based fairness is clearly unfair if this is the pricing method adopted, and VC/source-based fairness is superior. The application type also has a significant impact on this choice, since the application type determines the probability of having a large number of concurrent senders.

Summary and Conclusions. The above discussion shows that each type of fairness has its own merits and drawbacks, and the choice of the type of fairness to adopt relies on the application type, and pricing methods used. We believe that **source-based** fairness is the most preferred because the definition is a simple extension of point-to-point fairness definitions. In addition, it is the simplest to implement (refer to section 6), and does not suffer from beat-down problems as with flow-based solutions. Pricing can be based on sources in this case. We give in reference [7] a distributed algorithm that achieves source-based fairness, and we analyze the performance of the algorithm. VC/source-based fairness is appropriate for applications with large numbers of concurrent senders that price bandwidth by connection/session. The next two sections discuss the design and implementation of algorithms to compute the various types of fair allocations.

6 Design Issues

There are several ways to design multipoint-to-point ABR flow control algorithms. Each method offers a tradeoff in fairness, complexity, scalability, overhead, and response time. In this section, we examine the design of multipoint rate allocation algorithms with VP merging, VC merging, different types of accounting, and we discuss scalability issues.

- **Delay Sensitivity.** Some merge point algorithms wait for an FRM cell to be received before sending feedback. What are the implications of this on the scalability of the scheme? Will

the feedback delay grow with the number of levels of merge points? If each merge point must wait for the next FRM cell, the time to return a BRM cell can increase with the number of levels of the tree, which is an undesirable property. This is also dependent on the FRM and BRM cell rates, and their relationships during transient phases. Schemes that return the BRM cell received from the root, to the leaves which have sent FRM cells to the merge point since the last BRM cell was passed, are less sensitive to number of merge points.

- **Merging.** With *VC merge* implementations (refer to section 3), it is impossible to distinguish among the cells of different sources in the same multipoint-to-point VC (since the same VPI/VCI fields are used for all the cells of a VC on the same hop). Hence, switch traffic management algorithms must not rely on being able to determine the number or rates of *active sources* with VC merge (number and rates of active VCs and number and rates of active flows can still be determined). With *VP merge*, however, the VCI field is used to distinguish among cells of different sources in the same multipoint-to-point VC on the same hop. Hence, it is possible to determine the number and rates of *active sources* in such implementations, and perform any necessary per-source accounting operations. (This, however, may incur additional complexity and reduce scalability.)

- **Accounting.** All switch traffic management algorithms need to use some registers for storing the values they need to compute the rate allocations. Some of these values are stored for each input port, and some for each output port. Other algorithms use per-VC accounting, per-source accounting, or per-flow accounting. With multipoint-to-point VCs, per-VC accounting, per-source accounting, and per-flow accounting are no longer equivalent (they are equivalent for point-to-point scenarios). This leads to a set of interesting problems. For example, some algorithms store the value of the current cell rate (CCR) indicated in FRM cells, and later use it for rate computation, while others use the CCR value from BRM cells directly in rate computation to avoid O(N) storage. But the CCR value is actually per-source (and the sources cannot be distinguished with VC merge). We have shown in [7] that using the CCR field from BRM cells can cause unfairness to upstream sources. Further, some algorithms also attempt to measure the source rates, or distinguish overloading and underloading sources (e.g., MIT scheme, UCSC scheme, and basic ERICA). These schemes do not perform as expected in

the multiple sender case. In general, *per-source accounting is infeasible with VC merge, while per-VC accounting must account for the VC as a whole (even if its traffic is coming from different ports), and per-flow accounting must distinguish both input ports and VCs.*

- **Bi-level operation.** For point-to-point and point-to-multipoint connections, and for multipoint-to-point connections with source-based fairness, the switch computes the rate allocations it can support based on available ABR capacity, and indicates these allocations in the BRM cells (only if they are less than the allocations computed by downstream switches, as indicated in the ER field of BRM cells). This suffices for these situations since the algorithm operates at the source level only, and all sources at a bottleneck are allocated equal rates. With VC/source, flow, and VC/flow-based fairness, however, switches may need to first estimate the maximum available capacity for the VC/flows (possibly using the ER field in BRM cells coming from downstream, and the rate allocation algorithm applied at the link level), and then sub-divide this capacity among the senders/flows. Per-VC or per-flow accounting may be used for the sub-division operation. We call this bi-level operation. Alternatively, weights can be used to scale the rate allocations, but the weights depend on the number of senders/flows in the multipoint VC/port, which is a variable quantity.

7 Summary and Concluding Remarks

There are several issues to be resolved in ATM multipoint communication, including devising a scalable method for merging traffic from multiple senders, and resolving traffic management issues. ATM-ABR should be supported, as ABR provides good performance for both data and real-time applications, because of its loss and rate guarantees and delay control, as well as its efficient utilization of bandwidth and buffering.

Traffic management for multipoint connections may be implemented differently in VC merge and VP merge implementations. VP merge uses the VCI field to distinguish among different sources in the same multipoint VC, while VC merge does not distinguish sources, and implements packet-level buffering at the merge points.

Four different types of fairness can be defined for multipoint-to-point connections:

1. **Source-based fairness,** which divides bandwidth fairly among active sources as if they were sources in point-to-point connections, ignoring group memberships.

2. **VC/source-based fairness,** which first gives max-min fair bandwidth allocations at the VC level, and then fairly allocates the bandwidth of each VC among the active sources in this VC.

3. **Flow-based fairness,** which gives max-min fair allocations for each active flow, where a flow is a VC coming on an input link. Formally,

   $NumFlows_j$, $j \in OutputPorts =$

   $\forall i, i \in InputPorts, \sum_i$ Number of active VCs coming on port i

   and being switched to port j

4. **VC/flow-based fairness,** which first divides the available bandwidth fairly among the active VCs, and then divides the VC bandwidth fairly among the active flows in the VC.

Since sources, VCs, and flows are equivalent for point-to-point connections, but different for multipoint-to-point connections, it is important to note the differences between the per-source, per-VC and per-flow accounting. Per-source accounting cannot be performed in VC merge implementations. Per-flow accounting must distinguish VCs and input ports, while per-VC accounting must combine the VC information coming from different input ports.

Modifications are necessary for switch algorithms to implement each of the four types of fairness. For source-based fairness (the simplest), algorithms operating with VC merge should not perform source-level accounting, and must only use information supplied in the RM cells, in addition to aggregate measurements of load, capacity and queuing delays. VC/source, flow and VC/flow fair allocations require bi-level operation or use of weights.

While the fairness definitions are already included as baseline text in the living list of the ATM Forum traffic management working group, it is essential to continue this work to define the desirable forms of fairness, and extend current switch traffic management algorithms for multipoint connections. Extensive performance analysis is also crucial to examine the fairness, complexity, overhead, transient response, delays, and scalability tradeoffs involved. Multipoint-to-multipoint connections can be supported by combining branch point algorithms (e.g., [8]) with merge point

algorithms (e.g., [7]).

This work is also tied to the fairness issues in Internet multicast protocols. Again the fairness, reliability and congestion control method to adopt is application specific. The inter-group and intra-group fairness notions discussed in this paper need to be precisely defined and measured for Internet traffic, and protocols need to be developed to provide multicast services that achieve fairness. This topic is currently gaining considerable attention at the IETF, with the increasing popularity of group collaboration applications.

Acknowledgments

This research was sponsored in part by Rome Laboratory/C3BC Contract # F30602-96-C-0156.

References

[1] G. Armitage. Support for multicast over UNI 3.0/3.1 based ATM networks. Request for Comments 2022, November 1996.

[2] S. Bhattacharyya, J. Kurose, D. Towsley, and R. Nagarajan. Efficient rate-controlled bulk data transfer using multiple multicast groups. In *Proceedings of IEEE INFOCOM '98*, April 1998.

[3] D. Cavendish and M. Gerla. Rate based congestion control for many-to-many multicast ABR traffic. In *Proceedings of SPIE '98 Conference on Performance and Control of network systems II*, volume 3530, pages 122–130, November 1998.

[4] S. Cheung, M. Ammar, and X. Li. On the use of destination set grouping to improve fairness inmulticast video distribution. In *Proceedings of IEEE INFOCOM '96*, volume 2, pages 553–560, March 1996.

[5] D. DeLucia and K. Obraczka. Multicast feedback supression using representatives. In *Proceedings of IEEE INFOCOM '97*, April 1997.

[6] C. Diot, W. Dabbous, and J. Crowcroft. Multipoint communication: A survey of protocols, functions and mechanisms. *IEEE Journal on Selected Areas in Communications*, 15(3):277–290, April 1997.

[7] S. Fahmy, R. Jain, R. Goyal, and B. Vandalore. A switch algorithm for ABR multipoint-to-point connections. ATM Forum/97-1085R1, December 1997.

[8] S. Fahmy, R. Jain, R. Goyal, B. Vandalore, S. Kalyanaraman, S. Kota, and P. Samudra. Feedback consolidation algorithms for ABR point-to-multipoint connections. In *Proceedings of the IEEE INFOCOM*, volume 3, pages 1004–1013, March 1998.

[9] S. Floyd, V. Jacobson, S. McCanne, C-G Liu, and L. Zhang. A reliable multicast framework for light-weight sessions and application level framing. In *Proceedings of ACM SIGCOMM '95*, pages 342–356, October 1995.

[10] A. T. M. Forum. The ATM forum traffic management specification version 4.0. ftp://ftp.atmforum.com/pub/approved-specs/af-tm-0056.000.ps, April 1996.

[11] E. Gauthier, J.-Y. Le Boudec, and P. Oechslin. SMART: A many-to-many multicast protocol for ATM. *IEEE Journal on Selected areas in Communications*, 15(3):458–472, April 1997.

[12] M. Grossglauser and K. K. Ramakrishnan. SEAM: Scalable and efficient ATM multipoint-to-multopoint multicasting. In *Proceedings of the IEEE INFOCOM*, April 1997.

[13] J. Jaffe. Bottleneck flow control. *IEEE Transactions on Communications*, COM-29(7):954–962, July 1981.

[14] R. Jain. Congestion control and traffic management in ATM networks: Recent advances and a survey. *Computer Networks and ISDN Systems*, October 1996.

[15] R. Jain, S. Kalyanaraman, S. Fahmy, R. Goyal, and S. Kim. Source behavior for ATM ABR traffic management: An explanation. *IEEE Communications Magazine*, November 1996.

[16] M. Jeffrey. Scope, concepts and issues for the new multiway BOF. ATM Forum/96-0628, June 1996.

[17] T. Jiang, M. Ammar, and E. Zegura. Inter-receiver fairness: a novel performance measure for multicast ABR sessions. In *Proceedings of the ACM SIGMETRICS*, pages 202–211, June 1998.

[18] S. Komandur and D. Mosse. SPAM: A data forwarding model for multipoint-to-multipoint connection support in ATM networks. In *Proceedings of the sixth international conference on computer communications and networks (ICCCN '97)*, pages 383–388, September 1997.

[19] X. Li, S. Paul, and M. Ammar. Multi-session rate control for layered video multicast. In *Proceedings of SPIE '99 Conference on Multimedia Computing and Networking*, San Jose, CA, 1999.

[20] J. Lin and S. Paul. RMTP: A reliable multicast transport protocol. In *Proceedings of the IEEE INFOCOM*, volume 3, pages 1414–1424, March 1996.

[21] S. McCanne, V. Jacobson, and M. Vetterli. Receiver-driven layered multicast. In *Proceedings of ACM SIGCOMM '96*, pages 117–130, August 1996.

[22] W. Moh. On multicasting ABR protocols for wireless ATM networks. In *Proceedings of the International Conference on Network Protocols (ICNP) '97*, 1997.

[23] W. Moh and Y. Chen. Design and evaluation of multipoint-to-point multicast flow control. In *Proceedings of SPIE '98 Conference on Performance and Control of network systems II*, volume 3530, pages 143–154, November 1998.

[24] W. Ren, K-Y Siu, and H. Suzuki. On the performance of congestion control algorithms for multicast ABR service in ATM. In *Proceedings of IEEE ATM'96 Workshop, San Francisco*, August 1996.

[25] W. Ren, K-Y Siu, and H. Suzuki. Multipoint-to-point ABR service in ATM networks. In *Proceedings of the International Conference on Communications, ICC'97, Montreal*, June 1997.

[26] W. Ren, K-Y Siu, H. Suzuki, and M. Shinohara. Multipoint-to-multipoint ABR service in ATM. *Journal of Computer Networks a nd ISDN Systems*, 30(19):1793–1810, October 1998.

[27] L. Roberts. Rate based algorithm for point to multipoint ABR service. ATM Forum/94-0772R1, November 1994.

[28] E. Spiegel. Multipoint connection support in PNNI 2.0. *ATM Forum Perspectives, IEEE Network*, Nov/Dec 1997.

[29] R. Talpade, G. Armitage, and M. Ammar. Experience with architectures for supporting IP multicast over ATM. In *Proceedings of IEEE ATM'96 Workshop, San Francisco*, August 1996.

[30] J. Turner. Extending ATM networks for efficient reliable multicast. In *Proceedings of Workshop on Communication and Architectural Support for Network-based Parallel Computing*, February 1997.

[31] B. Vandalore, S. Fahmy, R. Jain, R. Goyal, and M. Goyal. A definition of general weighted fairness and its support in explicit rate switch algorithms. In *Proceedings of the Sixth International Conference on Network Protocols 1998 (ICNP '98)*, pages 22–30, ctober 1998. http://www.cis.ohio-state.edu/~jain/papers/icnp98_bv.htm.

[32] R. Venkateswaran, C. S. Raghavendra, X. Chen, and V. Kumar. Support for multiway communications in ATM networks. ATM Forum/97-0316, April 1997.

[33] L. Vicisano, J. Crowcroft, and L. Rizzo. TCP-like congestion control for layered multicast data transfer. In *Proceedings of IEEE INFOCOM '98*, volume 3, pages 996–1003, March 1998.

[34] H. Wang and M. Schwartz. Achieving bounded fairness for multicast and TCP traffic in the Internet. In *Proceedings of ACM SIGCOMM '98*, September 1998.

[35] I. Widjaja and A. Elwalid. Performance issues in VC-merge capable switches for IP over ATM networks. In *Proceedings of the IEEE INFOCOM*, volume 1, pages 372–380, March 1998.

[36] T. Wong, T. Henderson, S. Raman, A. Costello, and R. Katz. Policy-based tunable reliable multicast for periodic information dissemination. In *Proceedings of the third International Workshop on Satellite-based Information Services*, 1998.

All our papers and ATM Forum contributions are available through http://www.cis.ohio-state.edu/~jain/